YRKESHÖGSKOLAN NOVIA

RFID Agent Game

Utvecklingen av ett HTML5-spel

Markus Silvennoinen

Examensarbete för Tradenom-examen

Utbildningsprogrammet för Informationsbehandling

Raseborg 2014

EXAMENSARBETE

Författare: Markus Silvennoinen

Utbildningsprogram och ort: Informationsbehandling, Raseborg

Handledare: Rolf Gammals

Titel: RFID Agent Game – Utvecklingen av ett HTML5-spel

Datum: 9.5.2014 Sidantal: 47 Bilagor: 14

Abstrakt

Arbetets uppdrag är att utveckla ett webbaserat spel åt beställaren, Nordic ID, utgående från deras grundidé. Syftet med spelet är att öka antalet besökare på företagets webbplats, och således antalet potentiella kunder. Som teknologi valdes HTML5, som fortfarande var under utveckling. PHP-skript hanterar aktiviteterna på serversidan. Data lagras i textfiler och för det mesta i XML-format, eftersom beställarens MySQL inte hade PHP-stöd.

Arbetet behandlar produkten RFID Agent Game, samt dess utveckling. Det kan även ge en uppfattning om hur HTML5-spelutveckling kan ske överlag. Med finns JavaScript källkod, som normalt är förvrängd eller krypterad. PHP-koden syns inte i webbläsaren, och hålls som en hemlighet för säkerhetens skull, men dess funktionalitet beskrivs.

Uppdraget var utmanande, och kraven höga. Spelet behöver ännu finslipas efter att den skriftliga delen av arbetet är färdigt. Nordic ID har varit nöjd med spelets demo-versioner hittills, och det som snart blir den slutliga versionen ser redan lovande ut. Nordic ID har i planerna att nämna mig och spelet i deras kundtidning.

Språk: Svenska Nyckelord: HTML5, PHP, spel, Nordic ID, RFID, Bubble Shooter

---

OPINNÄYTETYÖ

Tekijä: Markus Silvennoinen

Koulutusohjelma ja paikkakunta: Tietojenkäsittely, Raasepori

Ohjaajat: Rolf Gammals

Nimike: RFID Agent Game – HTML5-pelin kehittäminen

Päivämäärä: 9.5.2014 Sivumäärä: 47 Liitteet: 14

Tiivistelmä

Opinnäytetyön tehtävä on kehittää web-pohjainen peli Nordic ID:lle. Peli perustuu heidän ydinajatukseen. Pelin tavoitteena on kasvattaa heidän kotisivun kävijämäärää, ja siten mahdollisten asiakkaiden määrää. Pelin kehitystekniikaksi valittiin HTML5, joka oli vielä itsessään kehitteillä. PHP-skriptit käsittelevät palvelinpuolen toimintoja. Tietojen tallentamiseen käytetään tekstitiedostoja ja XML:ää, sillä Nordic ID:n MySQL ei tue PHP:tä.

Työ käsittelee tuotteen RFID Agent Game:in, sekä sen kehittämisprosessin. Se voi myös antaa käsityksen siitä, miten HTML5 pelin kehittäminen voi tapahtua yleisesti. Luettavissa on myös JavaScript-lähdekoodia, joka on tyypillisesti hämärretty. PHP-koodi ei ole luettavissa selaimesta, ja turvallisuussyistä se pysyy salassa, mutta sen toimivuus on kuvattu.

Tehtävä oli haastava, ja vaatimukset korkeat. Vielä kirjallisen osuuden valmistuttua, peli on pientä hienosäätöä vailla. Nordic ID on ollut tyytyväinen pelin demoversioihin tähän asti, ja tuleva lopullinen versio näyttää jo lupaavalta. Nordic ID aikoo nimetä minut ja pelin heidän lehdessään.

Kieli: Ruotsi Avainsanat: HTML5, PHP, peli, Nordic ID, RFID, Bubble Shooter

---

BACHELOR'S THESIS

Author: Markus Silvennoinen
Degree Programme: Business Information Technology
Supervisors: Rolf Gammals

Title: RFID Agent Game – The Development of an HTML5 Game

Date: 9 May 2014 Number of pages: 47 Appendices: 14

Summary

The assignment of this thesis is to develop a web based game for the principal, Nordic ID. It was made according to the base idea, which Nordic ID had made but not realized. The aim of the game is to increase the number of visitors on Nordic ID's website, and thus the number of potential customers. HTML5, which was still under development, were chosen as the development technology. The activity on the server side is handled by PHP scripts. Text files and XML are used for storing data, as the principals MySQL did not have PHP support.

This thesis deals with the product RFID Agent Game, as well as its development. It can also give an idea of how HTML5 game development can take place overall. It includes JavaScript source code that is otherwise obfuscated. PHP code cannot be viewed in the Browser, and for security reasons it's preserved as a secret, but its functionality is described.

The task was challenging, and its demands were high. The game still needs some small improvements after the written part of the thesis is finished. Nordic ID has been pleased with the game's demo versions so far, and what will soon become the final version is already looking promising. Nordic ID has plans to name me and the game in their magazine.

Language: Swedish Key words: HTML5, PHP, game, Nordic ID, RFID, Bubble Shooter

---

Innehållsförteckning

1 Inledning

1.1 Bakgrund och uppdrag

Detta examensarbete har utförts som ett utvecklingsprojekt åt Nordic ID. Jag blev erbjuden uppdraget under ett företagsbesök till deras kontor i Åbo våren 2013, då jag svarade på deras fråga att jag troligtvis kommer att tillverka spel efter utexamination. Uppdraget var att programmera ett webbaserat spel, *RFID Agent Game*, utgående från en grundläggande plan, som Nordic ID inte har haft möjlighet att förverkliga. Som kontaktperson fungerade Hanna Östman, grafisk designer. Hon ordnade all grafik som inte var behändigt att rita i HTML5. Credits och Instructions har Copywritern varit med och formulerat. Serveradministratorerna lägger spelet på deras webbplats. Själv har jag stått för programmeringen och strukturingen av spelet, och även musikstyckena och ljudeffekterna. Enligt uppdragsbeskrivningen kan spelet få mer material senare, så jag byggde även ett administrativt gränssnitt till spelet.

1.2 Syfte och målsättning

Målet med uppdraget är att skapa ett webbaserat spel, med syftet att öka besökarantalet på Nordic ID:s hemsida, och således antalet potentiella kunder. Bakom ligger även det personliga målet att utveckla kunskaperna inom webbaserad spelprogrammering.

1.3 Målgrupp

Målgruppen för det skriftliga arbetet är personer som är intresserade av HTML5-spelutveckling. IT-kunniga får mest ut av texten, eftersom den innehåller flera tekniska termer och kodsnuttar. Somliga personer på Nordic ID kan även ha nytta av detta dokument som en guide över hur spelet är uppbyggt, om det behöver vidareutvecklas senare.

1.4 Omfattning

Detta examensarbete omfattar en beskrivning av alla verktyg som användes vid tillverkandet av spelet, idéer och tips för HTML5-spelutveckling som uppstått under utvecklingen samt hur RFID Agent Game är uppbyggt. Arbetet har krävt mycket kodning, och således innehåller den flera programlistningar. Koden är kommenterad på engelska,

eftersom själva koden brukar vara på engelska. Koden brukar vara på engelska för att undvika problem med t.ex. skandinaviska tecken, och för att det förstås bättre internationellt. Eftersom Nordic ID har kontakter runt om i Norden och längre bort, kan det ha stora fördelar att koda och kommentera på engelska. Orsaken till att somliga listningar finns som bilagor är att de är för långa för att ingå i huvudtexten. De är längre än en sida.

Spelet är fullständigt programmerat av mig, Markus Silvennoinen. Somliga mindre kodsnuttar har anpassats från kod, som publicerats på Stackoverflow, W3Schools med mera, men det mesta har programmerats från grunden utan varken *Content Management System* (CMS) eller *pluginer*. Spelet är uppbyggt som en JavaScript plugin, som ritar spelet i ett angivet område på en webbsida med hjälp av en funktion.

1.5 Definitioner

RFID (Radio Frequency Identification): En streckkodslik teknologi som med radiovågor kan hitta och identifiera mikrochip som finns i t.ex. prislappar.

Content Management System: ett ramverk för hantering av en webbplats, t.ex. Drupal, Joomla och WordPress.

Plugin (även Plug-in): ett alternativt tilläggsprogram som brukar vara lätt att integrera i ett system eller program. Exempel på pluginer är VST-plugins för musikproduktion och jQuery-plugins för avancerade funktioner till webbplatsen. Exempel hittas på http://plugins.jquery.com/.

HTML DOM: Document Object Model är en nod-teknologi som används för att komma åt HTML-element med JavaScript. Ett element i HTML representeras av ett DOM-objekt. DOM kan beskrivas som en lista på alla element som finns i ett HTML-dokument. (W3Schools).

Rekursion: Inom programmering är rekursion en metod som syftar till sig själv. Detta är ett resurssnålt sätt att skapa loopar. Rekursion kan man få till stånd då man filmar en TV-ruta som visar filmningen i direktsändning, eller om man målar en tavla som föreställer sig själv i ramar. Resultatet är alltid upprepningar, vilka går att begränsa och ändra i programkoden. Här har vi ett exempel: Om du fortfarande inte vet vad rekursion innebär, läs *detta* stycke igen.

Motor: Inom programmering och program, är en motor ett verktyg med en eller flera funktioner. För att exemplifiera motsvarar ett flygplan ett dataprogram, och dess motorer dataprogrammets motorer. Motorer är mycket vanliga i spel, speciellt grafikmotorer. Det finns till och med stora spelmotorer som hanterar grafik, fysik, ljud med mera; allt i ett paket. Motorer kan vara funktioner, objekt, mjukvaruutvecklingspaket (SDK) och även utvecklingsplattformar såsom Flash och Blender.

2 Allmänt om RFID Agent Game

2.1 Spelet ur användarens perspektiv

RFID Agent Game, även kallat RFID Game, är en slags Bubble Shooter (www.**bubbleshooter**.net/), som har ett Nordic ID och RFID tema. Istället för att poppa bubblor, skannar man dem. Det finns dessutom så kallade element bland bubblorna. Lyckas man frigöra element, får man extra poäng. Bubblor skjuts från en RFID-läsare. En annan aspekt som avviker från den traditionella Bubble Shootern är att det finns en energimätare, och energi töms varje sekund. Energi fylls på då man skannar bubblor och frigör element. Under produktionen gick det även att fuska genom att skriva i Internetläsarens JavaScript-konsol ”energy=100;”, vilket förstås fyller energin till maximum. I slutversionen är det så lite man kan fuska att det inte är värt det. Endast skickliga hackers och personer som vet koden på serversidan kan komma på sätt att fuska åt sig High Score på ett lönsamt sätt. Även om man vet koden på serversidan, är det inte lätt att fuska, eftersom spelet är planerat så att även jag, som gjort spelet, inte ska klara av att fuska lönsamt.

Startmenyn låter användaren välja ett tema, ändra ljud och musik på eller av och storleken på spelet enligt skärm- eller fönsterstorlek, se de högsta poängen och läsa instruktioner och Credits, före spelet startas. Då man ändrar på storleken på spelet, kommer spelet att laddas om i ett skilt fönster i *scalingoptions.html*, som beskrivs lite mera i kap 4.1 Struktur. Inställningarna importeras som URI-parametrar. Om musiken är igång, stängs den i det äldre fönstret, för att det inte ska spela i två fönster samtidigt. I standardinställningarna är ljud på, musik av och storleken på spelet 480 x 480 pixel, vilket borde vara samma storlek som på föräldraelementet på Nordic ID:s websida.

I spelet finns fyra olika färgers bubblor, varav en är unik för varje tema. Även läsaren kan variera mellan teman. Ursprungligen har spelet haft tre olika teman, Grocery, Fashion och

Logistics, men det kan ha ändrat, eftersom administratörerna har ett gränssnitt för temahantering.

Spelarna kan sikta med musmarkören och skjuta med vänstra musknappen. Alternativt kan man använda piltangenterna vänster, höger och upp, eller numeriska 4, 6 och 8 för att rotera läsaren och skjuta. Det går även att skjuta med pekningar, men det är inte lika lätt.

Efter Game Over visas de högsta poängen och man har alternativen att spela igen, eller gå tillbaka till menyn. Om man nått de högsta poängen får man fylla i sitt namn, och är det fråga om första plats, får även e-postadress ifyllas då man kan vinna ett pris vid slutet av månaden om man hålls på första plats.

2.2 Tekniskt

Spelet har programmerats i tidig HTML5, inklusive JavaScript och CSS3. Dessutom används PHP för att läsa och skriva de högsta poängen samt att förhindra märkvärdigt fusk. I paketet ingår ett administrativt användargränssnitt, *Admin tools*, som består för det mesta av PHP. Med de administrativa verktygen kan man se de högsta poängen för den aktuella månaden samt för den förra. Orsaken till att de förra poängen syns är att Nordic ID har i planerna att utdela ett pris åt personen som har månadens högsta poäng. I listan finns även användares e-postadresser, så att Nordic ID kan fråga adressen dit de ska sända priset. Med verktygen kan man även tömma de högsta poängen, både för aktuella och tidigare, men det är inte nödvändigt eftersom allt nödvändigt sker automatiskt, då session_controller.php körs. Med verktygen kan man även hantera spelets teman. Det går att lägga till, editera och ta bort teman. Det är även möjligt att göra ett tema osynligt för spelets användare genom att inaktivera det. I kap 4 Design och funktionalitet går vi djupare in på det tekniska.

3 Val av verktyg

3.1 HTML5 vs Flash

Före jag skulle kunna välja verktyg effektivt, måste jag välja vilken teknologi som spelet skulle bygga på. Enligt uppdragsgivaren var det ingen skillnad om spelet var gjort i Flash eller HTML5. Jag valde HTML5, för att enligt somliga källor, såsom *Page, K.* (2013), skulle Flash hamna i skuggan av HTML5, och det har även större förutsättningar att

fungera direkt i mobila enheter. Dessutom hade jag visst intresse att lära mig denna nya teknologi, som egentligen inte avviker stort från det gamla bekanta HTML. Valet var inte alltför lätt heller, eftersom jag har mer erfarenhet av att göra spel i Flash än i HTML. Det närmaste till ett spel som jag gjort i HTML är en ”GIF-gubbe” som går fram och tillbaka genom att positionen och bilden ändras med JavaScript. Hade jag vid valet vetat att Flash går att konvertera till HTML5, skulle jag säkert ha valt Flash, men då hade jag gått miste om mycket lärdom, även objekt-orienterad JavaScript som intresserade mig vid det tillfället.

Då RFID Agent Game var nästan klart, hittade jag en ganska bra jämförelse mellan Flash och HTML5 på http://flashvhtml.com/. Denna sida innehåller, förutom en jämförelse, länkar till en Flash- och en HTML5-version av samma spel, och möjligheten för besökare att rösta på sin favoritteknologi. Vid tidpunkten detta skrevs hade HTML5 5700 röster och Flash 2400, vilket tillsammans med jämförelsen ger en bild av hur viktigt det är att spel fungerar i mobila enheter. HTML5 audio behöver ännu utvecklas för att nå samma nivå som Flash. Flera tycker även att HTML5 och Flash inte behöver konkurrera, utan att de snarare kan samverka. Utvecklarna på *Code* (Code Computerlove u.å) ville även konkret bevisa att teknologierna kan samverka genom att utveckla ett pong-spel som består till hälften av HTML5 Canvas och till hälften av Flash.

3.2 HTML5

Eftersom HTML5 är en relativt ny teknologi då detta skrevs, ska vi ta en genomgång. HTML5 skiljer sig från de tidigare versionerna genom att stöda ljud, video och häftig dynamisk ritning. Dessutom är märkningsspråket mer standardiserad än dess tidigare versioner, vilket gör programmeringen lättare och minskar på olikheterna mellan webbläsare. Tidigare sköttes ljud, musik och komplex dynamisk ritning av pluginer såsom Flash, men nu fungerar allt direkt i webbläsaren. Nytt är även element som stöder östasiatiska tecken och ”höger till vänster”-text, sektionselement, nya formatteringselement, nya datatyper till formulären, mikrodata samt förenklad syntax. Kanske det mest negativa är att center-taggen kommer att försvinna, och istället ska centreringen ske med CSS eller JavaScript (W3Schools 2014). (Shah, N. & Ortíz, G. 2013).

3.3 Teknologin på serversidan

Ett mål var att spelet skulle lätt gå att sätta igång på Nordic ID:s server, och detta kräver att spelet använder sådana teknologier som de redan har installerat. Som svar på förfrågan fick jag veta att de använder Linux, Apache, PHP och MySQL. De har möjlighet att uppdatera PHP:n till en lämplig version vid behov. MySQL-versionen är däremot äldre än PHP, så dessa teknologier kan knappast samverka. Istället för MySQL föreslog jag att spelet skulle lagra informationen i till exempel XML-filer. Detta gick de med på.

3.4 DreamWeaver

Som programmeringsverktyg har använts *Adobe DreamWeaver CS6*. DreamWeaver ger stöd vid programmering av HTML, ColdFusion, PHP, CSS, JavaScript, ActionScript, XML, ASP med mera.

3.5 XAMPP

För att testa spelet lokalt, har jag använt XAMPP. Det innehåller bland annat Apache och MySQL. I detta projekt behövdes endast Apache eller Tomcat köras, för att PHP och postande med *XMLHttpRequest* skulle fungera. Under testandet upptäcktes att Internet Explorer inte kan spela ljud på localhost. Enligt *dsportes* (2011) och flera andra källor, beror felet på tolkningen av filernas MIME-typ, och det kan fixas genom att konfigurera Tomcats eller Apaches specifikation av MIME-typer.

3.6 Internetläsare

För att testa spelet överhuvudtaget behövs förstås ett program som kan rita spelet utgående från HTML-koden. För att spelet ska fungera i alla moderna, mest kända webbläsare, är det lämpligt att använda dessa vid testningen. Vid tiden då detta skrevs, räknades som moderna webbläsare Internet Explorer 9-11, Firefox, Chrome, Opera och Safari (Nesbyte u.å.). Dessa har även bra webbutvecklingsverktyg.

3.7 Fruity Loops Studio

Fruity Loops Studio, ofta förkortad till FL Studio, är ett professionellt ljudediterings- och musiksekvenseringsprogram, som utnyttjar VST-plugin-teknologi. Alla ljudeffekter och bakgrundsmusik i RFID Agent Game har producerats i FL Studio 10.

4 Design och funktionalitet

För att inte främja hackandet av RFID Agent Game, har jag i detta arbete döpt somliga filer till något annat än vad de egentligen heter. En person, som vet vad filerna heter, kan dock lätt komma fram till vilken fil texten syftar till.

4.1 Struktur

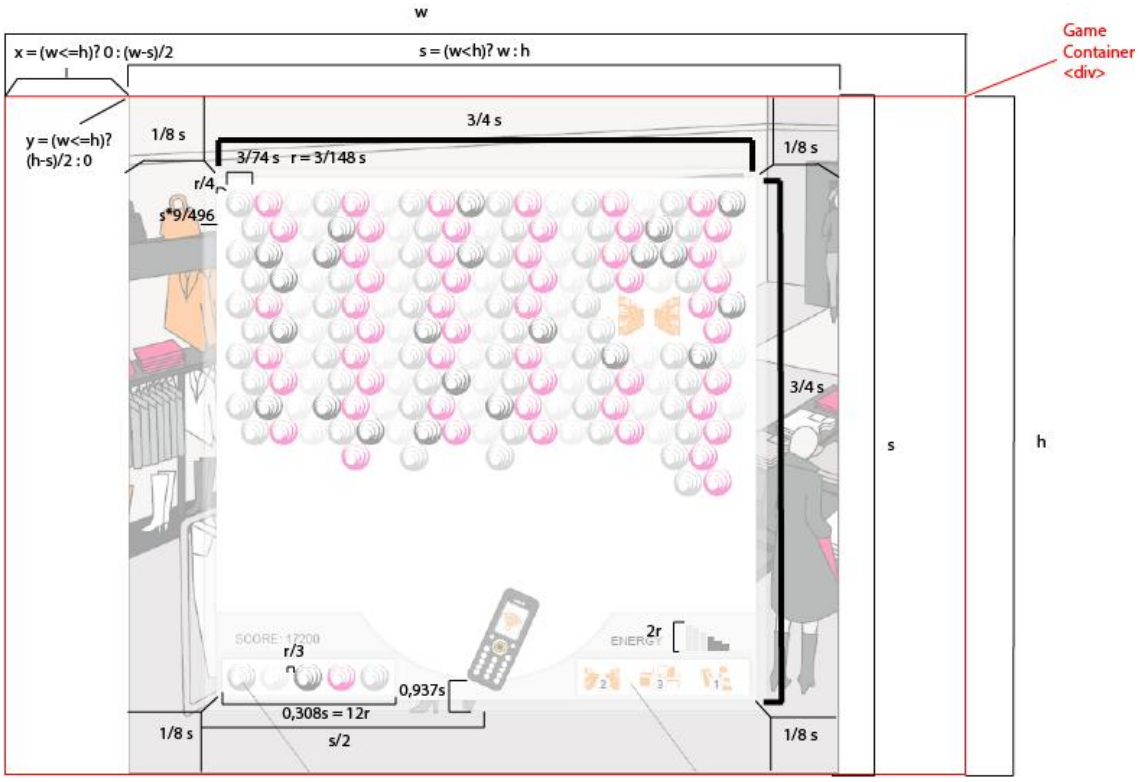
Hela spelet finns i mappen rfid_game. I denna mapp finns HTML-filer som anpassar spelet för olika ändamål. I *index.html* har vi en allmän version av spelet. *scalingoptions.html* skalar spelet enligt användarens önskemål i URL-förfrågan. Alternativen är *fw - Full Window*, som använder fönstrets maximalstorlek som mått, *fs - Full Screen*, som skalar enligt skärmens storlek och lämpar sig för *Full Screen Mode*, och *cw - Current Window*, som skalar enligt storleken på fönstret vid laddningen av spelet. Spelgränssnittet skriver dessa parametrar utgående från vad användaren valt i drop-down-menyn för skalning. Dessutom finns *embed.html*, som anpassar spelet för inbäddning på Nordic ID:s hemsida. Denna fil ritar spelet i storleken 480 x 480 pixel.

I rfid_game-mappen finns även en mycket viktig mapp, rfid_game_plugin, som innehåller all funktionalitet i spelet. De viktigaste filerna, eller programmen har egen rubrik. De övriga filerna är textfiler, som lagrar de högsta poängen i ett system med kommatecken som separator, och themes.xml, som lagrar spelets teman. Det finns även en textfil som lagrar månaden då spelet senast använts, eller mer specifikt session_controller.php som läser och uppdaterar filen. Månadsnumret används för att granska om månaden ändrats. Då månaden ändras ska månadens högsta poäng nollställas efter att de överförts till förra månadens lista över högsta poäng.

4.2 Design

En stor del av det grafiska fick Nordic ID:s grafiker, Hanna Östman sköta. Jag fick dock ge måtten åt henne några gånger, för att vi skulle undvika enorma dynamiska skalningar, som pixellerar bilden.

Spelet använder relativa mått. *Figur 1* visar hur måtten används. ”s” står för storlek eller skala, och syftar till sidlängden på det kvadratiska spelområdet. ”r” syftar till radien på bubblorna. ”s” värde beror på spelbehållarens mått; det får alltid det mindre värdet då bredden och höjden jämförs.



Figur 1. Spelets skalning

4.3 RFID Game Plugin JS

Filen rfid_game_plugin.js är kanske den viktigaste filen i hela spelet. Åtminstone innehåller den mest kod, över 2000 rader. Den laddas av HTML-filen som ska innehålla spelet. Med funktionen *insertRFIDGame* ritas spelet i ett element på sidan. Elementets storlek påverkar spelets storlek.

Koden är delad i sektioner. Den börjar med definitionen av *insertRFIDGame*. Sedan kommer definition av globala variabler, verktygsfunktioner, objekt och händelser. Det första som sker när filen laddas är att alla variabler och funktioner deklareras. En del variabler tilldelas ett värde direkt, men de flesta får ett värde först när *insertRFIDGame* anropas, eftersom flera värden är beroende av behållarelementets storlek.

4.3.1 Funktionerna insertRFIDGame och continueInsertion

Funktionen insertRFIDGame tar som argument ett DOM-objekt, i vilket spelet ska ritas, och som ett valfritt argument ett booleskt värde för huruvida musik ska spelas direkt eller inte. Som utgångsvärde ska musiken inte spela direkt. URL-förfrågan ”m=1” kan dock låta

musiken spela oberoende av argumentet. Detta är behändigt om spelet behöver laddas om och man vill bevara sin konfiguration.

I *insertRFIDGame* sker främst kalkylering och tilldelning av värden. Ursprungligen var det planerat att hela inläggsprocessen skulle ske i denna funktion, och så ser det ut utanför pluginen, men i verkligheten är den delad i fyra funktioner: *insertRFIDGame*, *loadThemes*, *getThemes* och *continueInsertion*. I slutet av insertRFIDGame anropas loadThemes, en verktygsfunktion som laddar spelets teman från en XML-fil med JavaScript objektet *XMLHttpRequest*. Detta objekt har händelsen *onreadystatechange*, vars uppgift är att anropa funktionen *getThemes* när filens innehåll har laddats framgångsrikt. I *getThemes* delas innehållet i en matris (array) för senare användning. I slutet av funktionen anropas *continueInsertion*. I *continueInsertion* laddas alla ljudfiler i HTML5-audio-taggar och alla fönster i spelet ritas, även om de flesta är osynliga i utgångsläget. Vid ritandet används attributet *innerHTML* på DOM-objekten. I funktionen ritas även modeller för de allmänna bubblorna. Funktionerna insertRFIDGame och continueInsertion finns i *bilaga 1*. De övriga får en skild genomgång framåt i texten.

4.3.2 Globala variabler

Det finns ett flertal globala variabler. De är delade i sektioner. De första variablerna är konstanter som tilldelas värden direkt. Här har vi till exempel den teoretiska startstorleken på spel-canvasen och gränser för somliga spelobjekt. Sedan kommer variabler som tilldelas värden då spelet ritas. Här har vi till exempel bubblans radie, som beror på behållarens storlek och fungerar vidare som ett mått för andra element. Efter det deklareras variabler som används för spelfunktionaliteten. Här har vi bland annat hastigheten på skjutna bubblor och temporära poäng. De sammanlagda poängen lagras på serversidan.

4.3.3 Verktygsfunktioner

I verktygsfunktionerna har vi granskning av bubbelkombinationer, bubbel- och elementgenerering, laddning av spelets teman, fuskgranskning via session_controller.php, spelloop, ritning, animering med mera.

Den första verktygsfunktionen *checkCompatibility* granskar om spelet fungerar i webbläsaren. Den granskar bara om webbläsaren kan använda HTML5 canvas och audio. Dessutom granskas om XMLHttpRequest fungerar. Om det inte gör det, antas webbläsaren vara en äldre version av Internet Explorer, som knappast är HTML5-kompatibelt.

Funktionen returnerar en sträng med HTML-kod som visas istället för spelet ifall att något är inkompatibelt. *Programlistning 1* visar hur funktionen är uppbyggd.

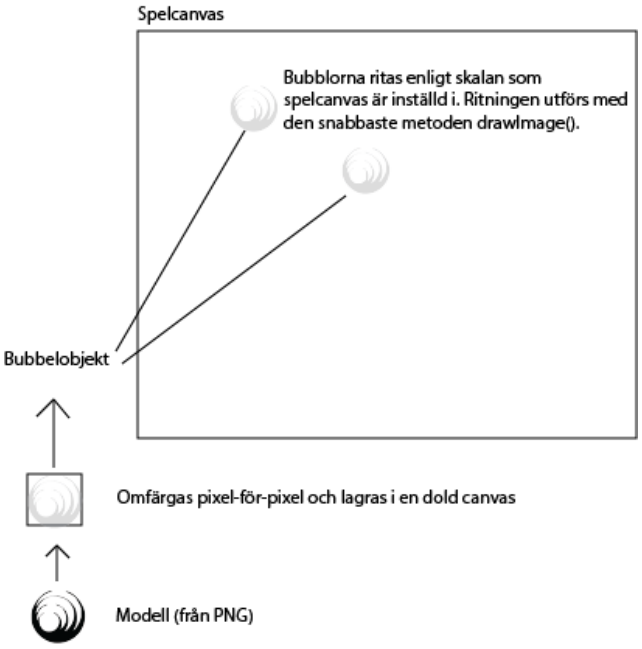
Listning 1. Programkod i JavaScript som granskar webbläsarens kompatibilitet.

```
// Check the Browser's support for RFID Game. Features that all
major browsers support are not included, except for basic HTML5
check.
function checkCompatibility() {
	var incompatibilities = "";
	var elem = document.createElement('canvas');
	if(!(elem.getContext && elem.getContext('2d')))
		incompatibilities += "HTML5 Canvas<br>";
	elem = document.createElement('audio');
	if(!elem)
		incompatibilities += "HTML5 Audio<br>";
	if(!window.XMLHttpRequest)
		incompatibilities += "XMLHttpRequest. (ActiveXObject,
found in older versions of Internet Explorer is ignored)<br>";
	return incompatibilities;
}
```

Funktionen *getCombinations* hittar alla bubblor som har samma färg och som är länkade till varandra, och returnerar en matris på dessa bubblor. Algoritmen går igenom alla närliggande bubblor, lagrar alla träffar som inte lagrats tidigare och utför samma procedur på de nya träffarna tills det inte finns nya träffar. Funktionens kod finns i *bilaga 2*.

Funktionen *getNonReleasedContent* returnerar en matris på alla bubblor och element som har förbindelse till övre kanten av spelplanen. Med hjälp av matrisen får man reda på vad som inte har förbindelse. Bubblor utan förbindelse poppas och element faller. Funktionen är mycket lik *getCombinations*, med skillnaden att den börjar söka kombinationer från övre spelkanten och att den istället för en viss färgs bubbla granskar om de närliggande positionerna innehåller objekt.

Funktionen *renderBubble* bidrar med resurssnålhet och effektivitet. Den används för att rita bubbelmodeller, som går snabbt att rita på spelcanvasen. Funktionen tar formen från en PNG-bild på en svart bubbla, varefter den kan ändra färgen på den och lagra den i en canvas. *Figur 2* visar hur det går till. Koden finns i *bilaga 3*.



Figur 2. Bubbelmodellanvändning

Funktionen *renderElement* har samma princip som *renderBubble*, men den använder istället direkta bilder och blir således flera rader kortare. Funktionen finns i *programlistning 2*.

Listning 2. Funktionen renderElement.

```
function renderElement(e_img_src) {
      var tix = eindex;
      e_models[eindex] = new Image();
      e_models[eindex].onload = function() {
            modelsLoaded++;
            e_models[tix]
      }
      e_models[eindex].src = e_img_src;
      eindex++;
}
```

Funktionen getURIParams extraherar och returnerar förfrågningsparametrarna i URL:en. Förfrågningsparametrarna används för att bevara konfigurationen även då spelet laddas om. *Programlistning 3* innehåller funktionen.

Listning 3. Funktionen getURIParams.

```
function getURIParams() {
	var url = document.URL.split("?");
	var obj = {m:2,s:2};
	if(url.length>1) {
		var params = url[1].split("&");
		if(params.indexOf("m=0")>-1) {
			obj.m = 0;
		}
		else if(params.indexOf("m=1")>-1) {
			obj.m = 1;
		}
		if(params.indexOf("s=0")>-1) {
			obj.s = 0;
		}
	}
	return obj;
}
```

Funktionerna loadThemes och getThemes laddar spelets teman från en XML-fil och organiserar dem i en matris av objekt. Objekten har i detta fall endast datatypen *Object*, till skillnad från de objekt som beskrivs i 4.2.4 Objekt, och innehåller temats namn, bakgrundsbildens adress, läsarens adress, bildadress för rubriken i drop-down-menyn, bildadress för rubrikens variant, den unika bubblans färg delad i rött, grönt och blått och ett Booleskt värde som indikerar om temat är aktiverat eller inte. Drop-down-menyns bilder lagras även i skilda matriser som används av drop-down-menyn. I *bilaga 4* finns källkoden till funktionerna.

Funktionen *generateContentOnRows* genererar bubblor och element till spelcanvasen. Funktionen tar som argument antal rader som ska fyllas. P.g.a. positioneringsorsaker är det säkrast att antal rader är ett jämt tal. För att element ska kunna genereras, måste åtminstone fyra rader genereras. Orsaken till detta är att undvika att element, som alltid kräver två rader, inte ska ramla direkt då de genereras. Koden till funktionen finns i *bilaga 5*.

För slumpmässigt skapande av bubblor och element har vi tre funktioner

```
	function randomBubble() {
		return new Bubble(randomInt(0,b_models.length));
	}
```

```
function randomElement() {
      return new Element(randomInt(0,e_models.length));
}
function randomInt(minv,amount) {
      return Math.floor(Math.random()*amount) + minv;
}
```

av vilka randomInt genererar ett slumpmässigt heltal och de övriga skapar själva objekten utgående från det slumpmässiga talet. Heltalet anger vilken typ av bubbla eller element det är frågan om; det används till och med för att hämta grafiken från modellmatriserna ”b_models” och ”e_models”.

Allt som finns i spelcanvasen ritas med *renderGame*. Funktionen kör även några längre kodsnuttar i skilda funktioner. Detta är funktionen som exekveras oftast i hela spelet (24-100 gånger per sekund), eftersom flera animationer är beroende av den. *Programlistning 4* visar hur funktionen är uppbyggd.

Listning 4. Ritning av spelet sker med renderGame.

```
function renderGame() {
      // Clear previous pixels on gameArea (but not game space
background)
      ctx.clearRect(0,0,osize+margin*2.1,osize+margin*2.1);
      // Draw the background for the incoming bubble line
      ctx.fillStyle = "#FFFFFF";
      ctx.fillRect(bline.x,bline.y,bline.w,bline.h);

      eMeter.DrawEnergy();
      elemCounter.Draw();
      sField.DrawScore();

      // Draw bubbles to canvas
      for(var i=0;i<bubbles.length;i++) {

      ctx.drawImage(bubbles[i].image,bubbles[i].x,bubbles[i].y);
      }
      // Draw elements to canvas
      for(var i=0;i<elements.length;i++) {

      ctx.drawImage(elements[i].image,elements[i].x,elements[i].y,b
r*5,br*4*Math.sin(abr));
```

```
	}
	drawReader();
}
```

Funktionen drawReader ritar läsaren utgående från värdena i läsarobjektet. Främst påverkas utseendet av läsarens bild och rotationsvinkel. Funktionen sparar kontextens utgångsinställningar, ställer in en ny plats för origo, vilket även fungerar som rotationspunkt, roterar kontexten och ritar läsarbilden på spelcanvasen. Efter det tilldelas kontexten utgångsinställningarna, vilket återställer origo och rotation. *Programlistning 5* visar funktionens uppbyggnad.

Listning 5. Funktionen drawReader.

```
function drawReader() {
	ctx.save();
	ctx.translate(reader.bx,reader.by);
	ctx.rotate(reader.a);
	ctx.drawImage(reader.image,-reader.w/2,-
reader.h,reader.w,reader.h);
	ctx.restore();
}
```

4.3.4 Objekt

Eftersom Objekt-Orienterad JavaScript avviker från det klassbaserade systemet, kan det vara värt att kort gå igenom det. JavaScript använder inte klasser utan definierar objekt direkt i *konstruktorer*. Dessutom går det att bygga på objekt i efterskott, eftersom språket är prototypbaserat. Det går även att definiera funktioner i funktioner, vilket är nödvändigt för att bygga upp objektets metoder. Egentligen är alla variabler objekt i JavaScript, liksom *Object*-datatypen i högnivåspråken. Objekt kan t.o.m. ärvas, men det kräver mer arbete än i det klassbaserade systemet som använder *extends* eller motsvarande nyckelord. (Trasviña, 2010).

I spelet är bubblor, element, läsare, rutnät, bubbelkö, poängfält, energimätare, elementstatistik, musikspelare och musikstycken definierade som objekt.

Bubbelobjektet är objektet med flest instanser i spelet. Det innehåller data för grafik, position och typ. *Programlistning 6* visar objektets konstruktor.

Listning 6. Bubbel-objektets konstruktor.

```
function Bubble(bubbleI,sx,sy) {
	if(sx===null) sx = 0;
	if(sy===null) sy = 0;
	this.x = sx;
	this.y = sy;
	this.gi = -1;
	this.gj = -1;
	this.type = "bubble";
	// Storage index
	this.i = bl;
	bl++;
	// Visual
	this.image = b_models[bubbleI];
	// Type of color expressed as a number
	this.colorType = bubbleI;
}
```

Parametern *bubbleI* är en siffra mellan noll och tre. Siffran representerar bubbeltypen, eller bubblans färg. Parametrarna *sx* och *sy* definierar en utgångsposition för bubblan i spelcanvasen. Utgångspositionen är (0,0), vilket i RFID Agent Game alltid är i övre vänstra hörnet, när något positioneras. Det enda undantaget är när läsar-objektet ska roteras, då canvasens origo ställs i läsarens bas. Variablerna *gi* och *gj* används som index-positionering i rutnätet. G:et står för ”Grid”. Variabeln *bl* står för ”bubbles length”, längden på bubbelmatrisen. Bubbelmatrisen innehåller alla bubblor som finns i spelet.

Bubbelobjektet har även metoderna ”Pop” och ”Remove”. Pop-metoden gör allt som bubblan ska göra då den skannas. Den spelar ”pop”-ljudeffekten, ökar poängen och energin samt tar bort bubblan med Remove-metoden. *Programlistning 7* visar metoderna.

Listning 7. Bubbelobjektets metoder.

```
// In the code "Pop" is used, but for the public, we call it
"scan", as that is what the RFID Reader does.
Bubble.prototype.Pop = function() {
	// Play pop sound and remove bubble
	if(soundEnabled) {
		sfx['pop'].pause();
		sfx['pop'].currentTime=0;
```

```
            sfx['pop'].play();
      }
      score++;
      energy+=0.6;
      if(energy>100) energy = 100;
      bPops++;
      this.Remove();
}
// Use this function to simply remove a bubble
Bubble.prototype.Remove = function() {
      // Update the theoretical index of each bubble
      for(var ix=this.i+1;ix<bubbles.length;ix++)
            bubbles[ix].i--;
      // Remove current bubble
      bubbles.splice(this.i,1);
      bl--;
}
```

Element-objektet avviker inte mycket från Bubblan, eftersom den följer samma logik. *Programlistning 8* visar Element-objektet i sin helhet.

Listning 8. Element-objektet.

```
function Element(elementI,sx,sy) {
      if(sx===null) sx = 0;
      if(sy===null) sy = 0;
      this.x = sx;
      this.y = sy;
      this.gi;
      this.gj;
      this.released=false;
      this.type = "element";
      // Storage index
      this.i = el;
      el++;
      // Visual
      this.image =new Image();
      this.image.src = e_models[elementI].src;
      // Type expressed as a number
```

```
	this.etype = elementI;
}
Element.prototype.Remove=function() {
	// Update the theoretical index of each element
	for(var ix=this.i+1;ix<elements.length;ix++)
		elements[ix].i--;
	// Remove the element itself
	elements.splice(this.i,1);
	el--;
}
```

Element hade tidigare en metod ”Release”, men p.g.a. funktionalitetshinder, måste det definieras som en skild funktion, *releaseElements*. Det var egentligen bra, för att funktionen blev ganska häftig när den fick som uppgift att animera alla fallande elementen samtidigt. *Programlistning 9* visar hur funktionen är uppbyggd.

Listning 9. Funktionen releaseElements

```
// Related to the Element objects
function releaseElements(elems,t) {
	if(t==null) t=br/2;
	//alert(t);
	for(var i=0;i<elems.length;i++) {
		elems[i].y+=t;
		if(elems[i].y>cs) {
			// Play element release sound
			if(soundEnabled) {
				sfx['erel'].pause();
				sfx['erel'].currentTime=0;
				sfx['erel'].play();
			}
			// Set score, energy and stats
			score+=10;
			energy+=10;
			if(energy>100) energy = 100;
			eRels++;
			elemCounter.Add(elems[i]);
			elems[i].Remove();
```

```
                elems.splice(i,1);
            }
        }
        renderGame();
        if(elems.length>0)

        setTimeout(function(){releaseElements(elems,t+br/8/fps);},100
0/fps);
}
```

Den rekursiva funktionen ovan får alla befriade elementen att accelerera neråt samtidigt. Först när ett element når nedre delen av spelområdet, ska poängeffekterna för det verkställas. Som argument tar funktionen en matris som innehåller de befriade elementen, samt en tid, som används för accelerationen.

Läsarobjektet innehåller endast attribut, såsom position, storlek, vinkel och grafik. *Programlistning 10* visar exakt hur läsarobjektet är uppbyggt. Objektet används i händelserna *mouseMove* och *onClick*. Dessa funktioner går vi igenom i 4.2.5 Händelser.

Listning 10. Läsarobjektet.

```
function Reader(img) {
        this.image = img;
        this.w = this.image.width*1/8;
        this.h = this.image.height*1/8;
        // bx and by are the canvas coordinates for the base of the
reader. These are used for e.g. rotation in canvas
        this.bx = osize/2+margin*1.2;
        this.by = osize+margin;
        // These are the base coordinates globally. These are used
when calculating the relative mouse position and getting the
correct rotation
        // for aiming purposes
        this.gbx = cx+cs/2;
        this.gby = cy+cs-margin*2;
        this.x = this.bx-this.w/2;
        //this.x = cs/2-this.w/6;
        this.y = this.by-this.h;
        this.a = 0;
}
```

Bubbelkö-objektet har kontroll över de inkommande bubblorna. Kön innehåller fem bubblor. När en bubbla skjuts iväg, eller skannas, flyttas den första bubblan i kön till funktionen som hanterar skjutningen. Samtidigt flyttas bubblorna i kön framåt i en animation som funktionen *Animate* åstadkommer. Koden till alla funktioner som ingår i Bubbelkö-objektet finns i *bilaga 6*.

Grid-objektet ser till att alla bubblor och element positioneras rätt på spelplanen. Objektet fungerar som ett osynligt rutnät, och dess sektioner kallas celler. Cellerna lagras i en tvådimensionell matris. Själva cellen är ett objekt som lagrar cellens koordinater och dess innehåll. Innehållet är en direkt referens till en bubbla eller ett element, så objektets metoder kan köras via cellen. Till exempel kan man med koden

```
Grid.cells[0][0].v.Pop();
```

poppa en bubbla som finns i vänstra övre hörnet. En tom cell kännetecknas av att *v* är *null*.

Grid-objektet har fem metoder. *PushContents* flyttar alla bubblor och element neråt i rutnätet, med ett angivet antal rader. Metoden granskar även om bubblor skulle hamna under nedre kanten, då spelet tar slut. Element kan inte nå nedre kanten utan att bubblor når den samtidigt, så i granskningen används endast bubblor. De övriga funktionerna är verktyg för konvertering mellan rutnätets indexering och pixelpositioner.

Källkoden till Grid-objektet och dess metoder finns i *bilaga 7*.

EnergyMeter är objektet som ritar energimätaren. Dess konstruktor är simpelt

```
function EnergyMeter() {
      this.energy = 100;
}
```

men dess metod DrawEnergy ritar energimätarens staplar. *Programlistning 11* innehåller koden till DrawEnergy.

Listning 11. Metoden DrawEnergy.

```
EnergyMeter.prototype.DrawEnergy = function() {
      var xoffset = osize*0.92;
      var yoffset = osize*0.87;
      var hoffset = br*2;
      var w = br/3;
```

```
	var space = Math.floor(w/10)+1;
	// The lower energylevel, the more energy there is left. This
is because the rectangles are indexed from left to right, from 0
to 5.
	// Energylevel(0) = full, EnergyLevel(6) = empty
	var eLevel = 6-Math.ceil(this.energy/100*6);
	// Grey
	var fill = "#CCCBCB";
	for(var i=0;i<6;i++) {
		if(i+1>eLevel)
			// Black
			fill = "#000000";
		/*/ Any effect when energy is low
		if(eLevel==5 && i==5) {
			// Red
			fill = "#FF0000";
		}
		*/
		// Any effect when out of energy (game over is
triggered in the gameLoop function)
		if(eLevel==6) {
			// Grey
			//fill = "#FF0000";
		}
		ctx.fillStyle = fill;
		var h = hoffset/6*(6-i)

	ctx.fillRect(xoffset+(w+space)*i,yoffset+hoffset/6*i,w,h);
	}
	ctx.fillstyle = "#000000";
	ctx.font="14px Arial";
	ctx.fillText("ENERGY",osize*0.77,osize*0.92);
}
```

ScoreField-objektet formatterar och ritar poängen. Objektet lägger till ett par nollor efter poängen, för att poängen ska se större ut än vad de är i lagret. 1 blir alltså 100. Enligt planen ska poängen gå att lagras med 24 bitar. Gränsen har avrundats till 10 miljoner, vilket tillsammans med de två dekorativa nollorna blir 1 miljard. Sannolikheten är ringa, men om någon lyckas nå 1 miljard poäng, stannar poängen vid 999 999 999 för att

poängen säkert inte ska överlappa andra visuella objekt. *Programlistning 12* visar källkoden till ScoreField-objektet.

Listning 12. ScoreField-objektet.

```
function ScoreField() {
	this.solidScore = 0;
}
ScoreField.prototype.DrawScore = function() {
	var str = "SCORE: "+(this.solidScore+score)+"00";
	if(this.solidScore+score<=0)
		str = "SCORE: 0";
	if(this.solidScore+score>=10000000)
		str = "SCORE: 999999999";
	ctx.font="14px Arial";
	ctx.fillStyle="#000000";
	ctx.fillText(str,2.5*margin,osize*0.92);
}
```

Objektet ElementCounter håller reda på antalet element som frigjorts. Den har även en metod som ritar siffrorna på spelcanvasen. *Programlistning 13* innehåller objektets konstruktor samt metoden Add.

Listning 13. ElementCounter-objektets konstruktor och metoden Add

```
function ElementCounter() {
	this.boxes = 0;
	this.cans = 0;
	this.clothes = 0;
	this.shelves = 0;
}
ElementCounter.prototype.Add = function(elem) {
	switch(elem.etype) {
		case 0:
			this.boxes++;
			break;
		case 1:
			this.cans++;
			break;
```

```
            case 3:
                  this.clothes++;
                  break;
            case 4:
                  this.shelves++;
                  break;
      }
}
```

Programlistning 14 innehåller ElementCounter-objektets metod Draw, som sköter om ritningen av siffrorna. Metoden ritar även en bild på varje element för att tydliggöra vilket tal hör ihop med vilket element.

Listning 14. ElementCounter-objektets Draw-metod.

```
ElementCounter.prototype.Draw = function() {
      // Define coordinates and draw white background
      ctx.fillStyle="#FFFFFF";
      var x = osize*0.68;
      var y = osize-margin*2-br/3;
      var w = 12*br;
      var h = 7/3*br;
      var space = br/2;
      var nums = [this.boxes,this.cans,this.clothes,this.shelves];
      ctx.fillRect(x,y,w,h);
      // Draw elements, circles and numbers
      for(var i=0;i<4;i++) {
            var x2 = x+(2.5*br+space)*i+br/4;
            var y2 = y+br/8;
            var w2 = 2.5*br;
            var h2 = 2*br;
            var cr = br/2;
            // Element
            ctx.drawImage(e_models[i],x2,y2,w2,h2);
            // Circle
            ctx.fillStyle="#F1F1F1";
            ctx.beginPath();
            ctx.arc(x2+w2/2,y2+br*1.5,cr,0,2*Math.PI);
            ctx.fill();
```

```
            // Number
            ctx.font="12px Arial";
            ctx.fillStyle="#000000";
            var str = (nums[i]>9)? nums[i] : " "+nums[i];
            ctx.fillText(str,x2+w2/3,y2+br*1.8);
      }
}
```

Nästa objekt är MusicPlayer, som kontrollerar uppspelningen av bakgrundsmusik. Den använder instanser av Track-objektet för att spela sekvenser definierade i programkoden. Metoden Play tar in som argument ett Track-objekt och sätter igång musiken. I *programlistning 15* finns objektets konstruktor och Play-metoden.

Listning 15. MusicPlayer-objektets konstruktor och Play-metod.

```
// An object that controls playback of music. It supports Intro,
Main loop and Outro.
// The intro is the music before loop start position, and outro is
the music after loop end position.
// The loop positions and repeats are defined in the Track-object
function MusicPlayer() {
      return this;
}
// Play a track
MusicPlayer.prototype.Play = function(track) {
      // timeout id used for cancelling the loop-event
      this.timer_id;
      // Stop any music that may be playing.
      if(this.music!=null)
            StopMusic();
      // If no track is specified and no track has been specified
earlier, the default track is played.
      if(track==null) {
            if(this.music==null) {
                  track = tracks[0];
            }
            // If no track is currently specified but a track has
been specified before, the previous track is played again.
            // This is used when re-enabling music.
            else {
```

```
                this.music[0].play();
                // Prepare the other audio instance for the
looping
                this.music[1].currentTime = this.loopStart;

      this.timer_id=setTimeout(LoopMusic,this.loopEnd*1000);
                return;
           }
      }
      this.track_id = track.id;
      this.track_name = track.name;
      this.music = track.music;
      this.loops = track.loops;
      this.corr = track.correction;
      this.loopStart = track.loopStart;
      this.loopEnd = track.loopEnd-this.corr;
      this.music[1].currentTime = this.loopStart;
      this.music[0].play();
      this.timer_id=setTimeout(LoopMusic,this.loopEnd*1000);
}
```

MusicPlayer har även tre funktioner som inte definierats som metoder, eftersom de används i själva objektet, och att köra metoder i själva objektet är besvärligt i JavaScript.

Funktionen LoopMusic används för att iterera en sektion i låten ett angivet antal gånger, varefter låten spelas till slut och nästa låt startas. Antalet iterationer är definierat i Track-objektet, som mPlayer innehåller. mPlayer är en instans av MusicPlayer. *Programlistning 16* innehåller källkoden till LoopMusic.

Listning 16. Funktionen LoopMusic.

```
function LoopMusic(count) {
      if(count==null)
           count = 0;
      if(count<mPlayer.loops) {
           // Play the other instance. It takes about 3 seconds
before the sound is played, so this function is executed corr
seconds in advance
           mPlayer.music[1-count%2].play();
           setTimeout(function(){
```

```
                // Pause the currently playing instance and
prepare it for eventual next iteration. (This is executed corr
seconds later)
                mPlayer.music[count%2].pause();
                mPlayer.music[count%2].currentTime =
mPlayer.loopStart;
          },mPlayer.corr*1000);

     mPlayer.timer_id=setTimeout(function(){LoopMusic(count+1);},(
mPlayer.loopEnd-mPlayer.loopStart)*1000);
     }
     // Play the file to its end and change track
     else
mPlayer.timer_id=setTimeout(changeTrack,(mPlayer.music[0].duration
-mPlayer.loopEnd)*1000);
}
```

Funktionen StopMusic stoppar musiken. Om musiken sätts igång igen startar den om den låt som avbröts. *Programlistning 17* visar koden bakom StopMusic.

Listning 17. Funktionen StopMusic.

```
function StopMusic() {
     clearTimeout(mPlayer.timer_id);
     // Make sure the both audio-instances are reset
     for(var i=0;i<2;i++) {
          mPlayer.music[i].pause();
          mPlayer.music[i].currentTime=0;
     }
}
```

Funktionen changeTrack byter låt till nästa på listan. *Programlistning 18* innehåller funktionens kod.

Listning 18. Funktionen changeTrack.

```
function changeTrack() {
     // Switch to next track
     var i = mPlayer.track_id+1;
     if(i>=tracks.length)
          i = 0;
     mPlayer.Play(tracks[i]);
```

```
}
```

Track-objektet används för att definiera musikstycken. *Programlistning 19* innehåller Track-objektet.

Listning 19. Track-objektet.

```
// Music Track object
function Track(name, music_dom, volume, loops, correction, loopStart, loopEnd) {
	if(volume==null) volume = 1;
	if(loops==null || loops<0) loops = 0;
	if(loopStart==null || loopStart<0) loopStart = 0;
	if(loopEnd==null || loopEnd>music_dom.duration) loopEnd = music_dom.duration;
	if(correction==null) correction = 0;
	this.name = name;
	// There must be two instances of the same audio, in order to create a seamless loop
	this.music = new Array(2);
	this.music[0] = music_dom;
	this.music[0].volume = volume;
	this.music[1] = music_dom.cloneNode(true);
	this.music[1].volume = volume;
	this.loops = loops;
	this.loopStart = loopStart;
	this.loopEnd = loopEnd;
	// Some looping sequences may require a correction in seconds, in order to play seamlessly.
	// This number is the same as the delay.
	// If the next iteration starts too late, e.g. there is silence, increase the correction. Too soon, decrease it.
	this.correction = correction;
	// The id may be used to play all tracks in order.
	this.id=tracks.length;
	return this;
}
```

4.3.5 Händelser

Denna sektion kan även innehålla funktioner som hänger ihop med själva händelse-funktionerna. I händelsesektionen hanteras musrörelser och klick, fusk, Game Over, knapptryckningar och tangenttryckningar.

Funktionen changeTheme anropas då spelets tema ändras i startmenyn. Koden till funktionen är

```
function changeTheme() {
	themePreview.src = themes[theme_sel.value].bg.src;
}
```

och det enda den gör är att den ändrar förhandsvisningen till det valda temat.

Funktionen startGame sätter igång spelet, då ”Play”-knappen används. Den utför de sista förberedelserna inför ett nytt spel genom att gömma startmenyn och eventuella scrollbars, sända en startsignal till serversidan, rita temats unika bubblas modell, läsaren och bakgrunden, generera innehåll på spelplanen, bygga upp bubbelkön, skala spelet enligt behållarens storlek och genom att definiera händelserna för spelinteraktion. Spelet kan inte starta helt direkt när denna funktion körs, eftersom klick-händelsen annars kan köras när man klickar på ”Play”-knappen, då en bubbla skjuts direkt iväg. Problemet har lösts med funktionen applyStart, som körs 20 millisekunder efter att startGame körts färdigt. Den sätter igång spelloopen, som beskrivs närmare i funktionen gameLoop, efter följande funktion. Funktionerna startGame och applyStart finns tillsammans i *bilaga 8*.

Funktionen updateScore granskar var femte sekund om poängen är rationella och lagrar dem i så fall på serversidan inför nästa granskning. Medan spelet är igång förväntar sig skriptet att funktionen körs med mindre än nio sekunders mellanrum, och om det inte gör det, tror programmet att spelaren försöker fuska, och poängen nollställs och nya poäng lagras inte under resten av spelet. Funktionens källkod finns i *programlistning 20*.

Listning 20. Funktionen updateScore.

```
// This function sends some information to the server-side every
fifth second. It detects rough cheating.
function updateScore() {
	if(gameOn) {
		var s_check_xhr = new XMLHttpRequest();
		s_check_xhr.onreadystatechange = function() {
```

```
                if (s_check_xhr.readyState == 4 &&
s_check_xhr.status == 200){
                    var resp = s_check_xhr.responseText;
                    //alert(resp);
                    var respA = resp.substring(10,resp.length-
11).split(",");
                    //alert(respA[0]+" "+respA[1]);
                    if(respA[0]==1) {
                        sField.solidScore = respA[1]*1;
                        //alert(respA[1]);
                        score = 0;
                        bPops = 0;
                        eRels = 0;
                    }
                    else resetGame("No Cheating!");
                }
            }

     s_check_xhr.open("POST","rfid_game_plugin/session_controller.
php",true);
            s_check_xhr.setRequestHeader("Content-
type","application/x-www-form-urlencoded");
            var
code=""+v+startGame.toString()+updateScore.toString()+gameLoop.toS
tring()+resetGame.toString()+playAgain.toString()+
            gameOver.toString()+bubbleHitTest.toString();
            var args =
"status=1&score="+encodeURIComponent(score)+"&pops="+encodeURIComp
onent(bPops)

     +"&rels="+encodeURIComponent(eRels)+"&energy="+encodeURICompo
nent(energy)+"&code="+encodeURIComponent(code);
            //alert(code);
            s_check_xhr.send(args);
            setTimeout(updateScore,5000);
     }
}
```

Funktionen gameLoop exekveras rekursivt 48 gånger i sekunden, eller så många gånger som variabeln fps (frames per second) är ställd till. Funktionen ansvarar för flera av

animationerna och timingen. Den lägger till innehåll på spelplanen med en frekvens, som ökar med poängen och tiden. Funktionens källkod är tillgänglig i *bilaga 9*.

Funktionen onKeyDown anropas då spelaren trycker ner en tangent. Då tar funktionen reda på om tangenten ska rotera läsar-objektet till vänster eller höger. Vänster- och högerpiltangenterna samt nummerpanelens fyra och sexa roterar läsaren. Funktionen onKeyUp, som kommer till nästa, kompletterar denna funktion. *Programlistning 21* innehåller källkoden till denna funktion.

Listning 21. Funktionen onKeyDown.

```
// Gets triggered when a key is down
function onKeyDown(e){

	var keynum;
	keyDown = true;
	if(window.event){ // IE
		keynum = e.keyCode;
	}
	else if(e.which){ // Netscape/Firefox/Opera

		keynum = e.which;
	}
	if(keynum == 39 || keynum == 102) {
		key = "right";
	}
	else if(keynum == 37 || keynum == 100) {
		key = "left";
	}
}
```

Funktionen onKeyUp anropas då en tangent släpps. Med denna funktion avslutas läsarens rotation. Den granskar även om piltangenten uppåt eller åttan släpps, då en bubbla ska skjutas iväg. *Programlistning 22* visar hur funktionen är uppbyggd.

Listning 22. Funktionen onKeyUp.

```
// Gets triggered when a key is released
function onKeyUp(e){

	var keynum;
	keyDown = false;
	key = "nodir";
	if(window.event){ // IE
		keynum = e.keyCode;
	}
	else if(e.which){ // Netscape/Firefox/Opera

		keynum = e.which;
	}
	// Up arrow or Number 8 key
	if(keynum==38 || keynum==104) {
		if(!shooting && !gamePaused && gameOn) {
			shooting=true;
			// Release the bubble from the incoming bubbles
and set the starting values of the previous indexes
			var b=bline.Release();
			prevI = grid.RowIFromY(b.y);
			prevJ = grid.ColumnJFromXI(b.x,prevI);

	shoot(b,v*Math.cos(reader.a+Math.PI/2),v*Math.sin(reader.a+Ma
th.PI/2));
		}
	}
}
```

Funktionen mouseMove anropas då musmarkören rör sig inom spelets behållare. Funktionen ändrar läsar-objektets rotation enligt musmarkörens position. *Programlistning 23* innehåller koden till funktionen.

Listning 23. Funktionen mouseMove.

```
// Gets triggered when the cursor moves inside the container
function mouseMove(evt) {
	//if(!shooting) {
	var mouseX = evt.clientX + 
document.documentElement.scrollLeft;
	var mouseY = evt.clientY + 
document.documentElement.scrollTop;
	if(evt.changedTouches) {
		mouseX = evt.changedTouches[0].clientX - 
document.documentElement.scrollRight;
		mouseY = evt.changedTouches[0].clientY - 
document.documentElement.scrollTop;
	}
	//var a = Math.atan2(mouseY-cy-10-reader.by,mouseX-cx-10-
reader.bx)+Math.PI/2;
	var a = Math.atan2(mouseY-reader.gby,mouseX-
reader.gbx)+Math.PI/2;
	// Increase the movability of the reader if you uncomment
	// a *=3;
	if(a<minAngle) a = minAngle;
	else if(a>maxAngle) a = maxAngle;
	reader.a = a;
	renderGame();
	/*/Debugging text
	ctx.font="24px Arial";
	ctx.fillStyle="#000000";
	ctx.fillText("a: "+Math.round(a*180/Math.PI)+"
vy:"+(Math.sin(a+Math.PI)),10,50);
	//ctx.fillText(a*180/Math.PI-90+" mx: "+(mouseX-cy)+" my:
"+(mouseY-cy),10,50);
	//*/
}
```

Funktionen onClick anropas då spelaren använder vänstra musknappen i spelbehållaren. Då ska en bubbla skjutas iväg. Funktionen finns i *programlistning 24*.

Listning 24. Funktionen onClick.

```
// Gets triggered if the user clicks inside the container
function onClick(evt) {
	if(!shooting && !gamePaused && gameOn) {
		shooting=true;
		// Release the bubble from the incoming bubbles and set the starting values of the previous indexes
		var b=bline.Release();
		prevI = grid.RowIFromY(b.y);
		prevJ = grid.ColumnJFromXI(b.x,prevI);

	shoot(b,v*Math.cos(reader.a+Math.PI/2),v*Math.sin(reader.a+Math.PI/2));
	}
}
```

Funktionen *shoot* skjuter iväg bubblan. I spelet kallas detta även för skanning. Funktionen tar som argument bubblan som ska skjutas iväg samt hastigheterna på x- och y-axeln. Bubblan kommer direkt från Bubbelkö-objektet. Funktionen är rekursiv och körs så många gånger i sekunden som variabeln fps är ställd till, för att animera bubblan. Den granskar även om bubblan rör kanterna av spelet, då riktningen på x-axeln ska inverteras, samt om bubblan träffar övre kanten eller ett annat objekt på spelplanen, då skjutningen ska avslutas. *Programlistning 25* visar funktionens konsistens.

Listning 25. Funktionen shoot.

```
function shoot(bubble,vx,vy) {
	bubble.x-=vx;
	bubble.y-=vy;
	renderGame();
	//check if the bubble collides with the horizontal edges of the area. Change its x-direction if it does.
	if(bubble.x>osize-margin) {
		vx = -vx;
		bubble.x=osize-margin;
	}
	if(bubble.x < margin) {
		vx = -vx;
```

```
            bubble.x=margin;
      }
      if(bubbleHitTest(bubble)) {
            if(!shooting)
                  resetGame("No Cheating!");
            shooting = false;
            renderGame();
      }
      else setTimeout(function(){shoot(bubble,vx,vy);},1000/fps);
}
```

Funktionen bubbleHitTest är massiv. Den tar in en bubbla som skjutits iväg och gör sex saker:

1. Den granskar om den skjutna bubblan träffar en annan bubbla eller ett element, då bubblan ska stanna, och bubblan stannar eftersom bubbleHitTest returnerar då sant till shoot-funktionen, som då förstår att sluta iterera.

2. Den positionerar bubblan enligt det dolda rutsystemet då bubblan stannat och lägger den med i Grid-objektet.

3. Den använder funktionen getCombinations för att ta reda på om bubblor ska poppas, och poppar sedan de eventuella kombinationerna. Ytterligare räknar den utgående från antalet kombinationer ut bonuspoäng.

4. Efter poppandet av bubblor, granskar funktionen om det finns bubblor som inte längre är länkade till övre kanten, och poppar dem.

5. Den samlar alla element som har frigjorts efter allt poppande, och sänder dem till releaseElements-funktionen.

6. Den utför det som beskrivs i punkterna 4 och 5 tills det inte sker förändringar på spelplanen.

Källkoden till bubbleHitTest-funktionen finns i *bilaga 10*. I koden är funktionen delad i fyra steg, och punkterna ett och två finns i första steget medan fem och sex finns i det sista.

Funktionen gameOver körs då spelet ska avbrytas. Den sänder ett meddelande till serversidan som säger att spelet är slut. Utan detta meddelande kan poäng inte registreras, eftersom det annars tolkas som fusk av skriptet på serversidan. Som svar på meddelandet får funktionen några värden, som berättar om spelaren har nått de 15 högsta poängen och i så fall vilken position hen har nått, och om det skett fusk. Utgående från värdena kan funktionen sedan rita eller låta bli att rita ett formulär för poängregistration. *Bilaga 11* visar hur funktionen är uppbyggd.

Funktionen showHighScore är uppbyggd så att den går att använda både i menyn och efter att spelet tar slut. Den skriver ut de högsta poängen i en tabell. Funktionen tar in två argument, *r* och *h*. Argumentet *r* står för ”result” och förväntas antingen vara tomt eller innehålla ett Booleskt värde som berättar åt funktionen att antingen skicka iväg användarnamn och e-post till servern, eller skippa det. Är *r* tomt, berättar det åt funktionen att de högsta poängen visas i menyn, och då ska det gå att skifta mellan alla tiders och månadens poäng, och återvändningen till menyn kräver inte återställning av spelets variabler. Har *r* däremot ett värde, visas poängen efter att spelet tagit slut och då finns alternativen att spela igen eller att återvända till menyn. Båda alternativen återställer de variabler som är nödvändiga, för att ett nytt spel ska kunna startas normalt. *Bilaga 12* innehåller källkoden till funktionen.

Funktionen showInstructions tar fram spelets instruktioner. Funktionen anropas då användaren klickar på menyknappen för instruktionerna. Funktionerna showOptions och showCredits följer samma princip, men för alternativ och tillverkare. *Programlistning 26* visar koden till de tre funktionerna.

Listning 26. Funktionerna showInstructions, showOptions och showCredits.

```
function showInstructions() {
      info.style.visibility="visible";
      info.style.zIndex=1;
      menuDisabler.style.opacity=0.4;
      menuDisabler.style.visibility="visible";
      mainMenu.style.zIndex=-1;
      showSBars();
}
function showOptions() {
      options_div.style.visibility="visible";
      menuDisabler.style.opacity=0.4;
```

```
	menuDisabler.style.visibility="visible";
	mainMenu.style.zIndex=-1;
}
function showCredits() {
	credits.style.visibility="visible";
	// z-indexing is required because of a scroll-bar visibility
bug in Opera.
	credits.style.zIndex=1;
	mainMenu.style.zIndex=-1;
	menuDisabler.style.opacity=0.4;
	menuDisabler.style.visibility="visible";
	showSBars();
}
```

Funktionen backToMenu är motsatsen till de tre funktionerna ovan. Den ser till att alla menyfönster är dolda och att huvudmenyn syns. *Programlistning 27* innehåller funktionen.

Listning 27. Funktionen backToMenu.

```
function backToMenu() {
	highscore.style.visibility="hidden";
	info.style.visibility="hidden";
	info.style.zIndex=0;
	options_div.style.visibility="hidden";
	credits.style.visibility="hidden";
	credits.style.zIndex=0;

	mainMenu.style.visibility="visible";
	mainMenu.style.zIndex=0;
	menuDisabler.style.visibility="hidden";
	hideSBars();
}
```

Funktionen playMuteMusic anropas då musiken ska ställas på eller av, t.ex. via checkboxarna i alternativen. *Programlistning 28* visar källkoden till funktionen.

Listning 28. Funktionen playMuteMusic.

```
function playMuteMusic(p) {
	var cb1=document.getElementById('music_cb1');
	var cb2=document.getElementById('music_cb2');
	if(p===1) {
		cb1.src=ebgr+'checkbox_checked.png';
		cb2.src=ebgr+'checkbox.png';
		if(!musicEnabled) {
			musicEnabled=true;
			mPlayer.Play();
		}
	}
	else if(p===0) {
		cb1.src=ebgr+'checkbox.png';
		cb2.src=ebgr+'checkbox_checked.png';
		if(musicEnabled) {
			musicEnabled=false;
			StopMusic();
		}
	}
	// In case of other solutions
	else {
		if(musicEnabled) {
			StopMusic();
			musicEnabled = false;

		}else {
			mPlayer.Play();
			musicEnabled = true;
		}
	}
}
```

Funktionen playMuteSFX fungerar på samma sätt som playMuteMusic, men den påverkar ljudeffekterna istället. *Programlistning 29* innehåller funktionens kod.

Listning 29. Funktionen playMuteSFX.

```
function playMuteSFX(p) {
	var cb1=document.getElementById('sound_cb1');
	var cb2=document.getElementById('sound_cb2');
	if(p==1) {
		cb1.src=ebgr+'checkbox_checked.png';
		cb2.src=ebgr+'checkbox.png';;
		if(!soundEnabled) {
			soundEnabled=true;
			sfx['erel'].play();
		}
	}
	else if(p==0) {
		cb1.src=ebgr+'checkbox.png';
		cb2.src=ebgr+'checkbox_checked.png';
		if(soundEnabled) {
			soundEnabled=false;
			sfx['erel'].pause();
			sfx['erel'].currentTime=0;
		}
	}
	// In case of other solutions
	else {
		if(soundEnabled) {
			soundEnabled = false;
		}else {
			soundEnabled = true;
		}
	}
}
```

Funktionen setSize lagrar värdet för storlekskonfigurationen för spelet i en variabel, så att spelet kan ritas om i en annan skala. Denna funktion kanske försvinner före den slutliga versionen av spelet är färdig, eftersom det finns vissa utmaningar med olika skärmstorlekar och centrering vid ändring av webbläsarfönstrets storlek. *Programlistning 30* innehåller funktionens källkod.

Listning 30. Funktionen setSize.

```
function setSize(sz) {
	switch(sz) {
		case 1: scaleMode="fw"; break;
		case 2:	scaleMode="fs"; break;
		case 3:	scaleMode="cw"; break;
		default: scaleMode="no";
	}
	for(var i=1;i<=4;i++) {
		document.getElementById('size_cb'+i).checked=false;
	}
	document.getElementById('size_cb'+sz).checked=true;
}
```

Funktionen optionsCheck granskar om det finns orsak till att ladda om spelet efter konfigurationen. Den enda orsaken hittills har varit att spelet ska ritas i en annan skala. Då formulerar funktionen konfigurationen till URL-parametrar, som spelet kommer att läsa då det laddas om. Spelet laddas sedan om i en ny flik, och i den gamla fliken mutas musiken om den är på. *Programlistning 31* innehåller funktionen.

Listning 31. Funktionen optionsCheck.

```
function optionsCheck() {
	// If there is a change in the size, the game is reloaded
into a new window
	if(scaleMode!="no") {
		var uri =
'm='+(musicEnabled?1:0)+'&s='+(soundEnabled?1:0)+'&size='+scaleMod
e;
		window.open('index.html?'+uri);
		playMuteMusic(0);
	}
	backToMenu();
}
```

Funktionen playAgain används då spelet tagit slut och spelaren antingen väljer att spela igen eller återvända till startmenyn. Funktionen används i båda fallen och det Booleska argumentet ”yes” innehåller spelarens beslut. Funktionen återställer alla nödvändiga variabler för att ett nytt spel ska kunna startas normalt, och ger ett värde åt variabeln reset,

som används främst i funktionen startGame för att undvika skalning på skalning med mera. *Programlistning 32* visar hur playAgain-funktionen är uppbyggd.

Listning 32. Funktionen playAgain.

```
function playAgain(yes) {
	// Reset variables
	menuDisabler.style.visibility="hidden";
	gameOn = false;
	shooting = false;
	score = 0;
	sField.solidScore = 0;
	elemCounter.boxes = 0;
	elemCounter.cans = 0;
	elemCounter.clothes = 0;
	elemCounter.shelves = 0;
	time = 0;
	pushCount = 0;
	bPops = 0;
	eRels = 0;
	energy = 100;
	// Clear grid
	for(var i=0;i<amountRows;i++)
		for(var j=0;j<bubblesPerRow;j++)
			grid.cells[i][j].v=null;
	bubbles = new Array();
	elements = new Array();
	bl = 0;
	el = 0;
	highscore.style.visibility="hidden";
	// Set restart value, to prevent bugs caused by another
rendering
	restart = 1;
	if(yes)
		startGame();
	else {
		restart = 2;
		backToMenu();
	}
```

```
}
```

Funktionen resetGame anropas då spelet märker fusk. Tidigare användes den även istället för playAgain-funktionen. Funktionen meddelar skriptet på serversidan att spelet återställs, varefter den visar ett eventuellt meddelande, som den tar in som argument, och laddar om hela spelet. Konfigurationen bevaras ändå med hjälp av URL-parametrar. *Programlistning 33 innehåller* funktionen.

Listning 33. Funktionen resetGame.

```
function resetGame(msg) {
	// Reset values
	var s_check_xhr = new XMLHttpRequest();
	s_check_xhr.open("POST","rfid_game_plugin/session_controller.
php",true);
	s_check_xhr.setRequestHeader("Content-type","application/x-
www-form-urlencoded");
	var
code=""+v+startGame.toString()+updateScore.toString()+gameLoop.toS
tring()+resetGame.toString()+playAgain.toString()+gameOver.toStrin
g()
	+bubbleHitTest.toString();
	var args =
"status=5&score=0&pops=0&rels=0&energy=0&code="+encodeURIComponent
(code);
	s_check_xhr.send(args);


	// Show eventual message
	if(msg!=null)
		if(msg.trim()!="")
			alert(msg);
	// Refresh the session in a way that is probably suitable for
the embedded version too
	var url = window.location.href;
	// Remove any details in the url that may cause a conflict
	var i = url.indexOf("#");
	var j = url.indexOf("?");
	if(i>-1) {
		if(j>-1) url = url.substring(0,i)+url.substr(j);
		else url.substring(0,i);
	}
```

```
	// Split arguments and remove any conflicts
	var urlA;
	var uri =
"m="+(musicEnabled?1:0)+"&s="+(soundEnabled?1:0)+"&size="+scaleMod
e;
	if(url.indexOf("?")>-1) {
		urlA = url.split("?")[1].split("&");
		for(var k=0;k<urlA.length;k++) {
			if(urlA[k].indexOf("m=")<0 &&
urlA[k].indexOf("s=")<0 && urlA[k].indexOf("size=")<0)
				uri += "&"+urlA[k];
		}
	}
	if(j>-1) url = url.substring(0,j);
	url = url+"#"+container.id+"?"+uri;
	//url = url.substring(0,j)+"?"+uri;
	//alert(url);
	window.location.href = url;
	location.reload();
}
```

4.3.6 Motorer

Den enda programmotorn som används i spelet är DropDownMenu. Motorn är i objektformat. Motorn har programmerats av mig och speciellt med tanke på spelet. Idén med den egna ”drop down”-menyn är att den går fullständigt att anpassa till utseendet, till skillnad från det inbyggda HTML ”select”-elementet. DropDownMenu tar faktiskt emot bilder till valalternativen, och ger möjlighet att ha bilder för ”hover-on”-effekter. Det märks att motorn är skräddarsydd för RFID Agent Game, men det finns attribut i den som är gjorda med tanke på allmänt bruk. Källkoden till DropDownMenu-motorn finns i *bilaga 13*.

Det sista i rfid_game_plugin.js är en bortkommenterad grafikmotor, vars syfte är att bevara bra kvalitet på bilder då de skalas i HTML5. Koden är kopierad från ett Stackoverflow-svar skriven av någon med användarnamnet ”syockit”. Koden hade små syntaxfel, men de var lätta att redigera. Motorn baserar sig på ett öppet grafikediteringsprogram och använder dess avancerade matematiska formler vid skalningen av bilder. Motorn används inte i spelet, eftersom det inte producerar önskade resultat och motorn kräver onödigt mycket

processorarbete. Motorn och den exakta adressen till Stackoverflow-frågan finns i *bilaga 14*, för dem som är intresserade.

4.4 Session Controller PHP

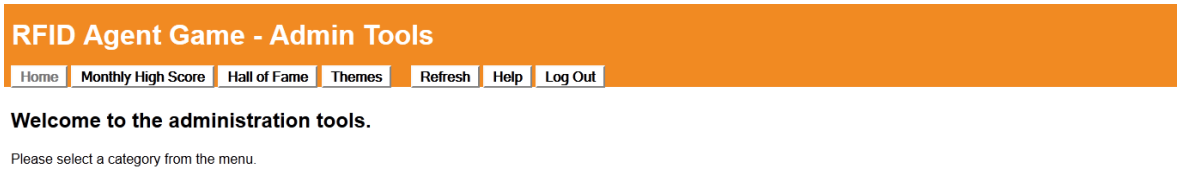
Om rfid_game_plugin.js är den viktigaste filen, är session_controller.php definitivt den nästviktigaste. Denna fil sköter allt automatiskt på serversidan som har med spelet att göra. De två viktigaste uppgifterna som programmet har är att upptäcka fusk och hantera de högsta poängen. Dessutom minns den spelarens namn och e-post under sessionen, och sänder dessa till RFID Game Plugin som kan fylla i dessa färdigt i fälten för poängregistrering. Programmet tar emot några argument från rfid_game_plugin.js, som berättar hur mycket poängen ökat och i vilket skede i spelet användaren befinner sig i. Som respons berättar programmet om förhållandena är giltiga och kan även ge ytterligare argument, såsom de nuvarande poängen och HTML, beroende på förhållandet.

4.5 Administrative Tools PHP

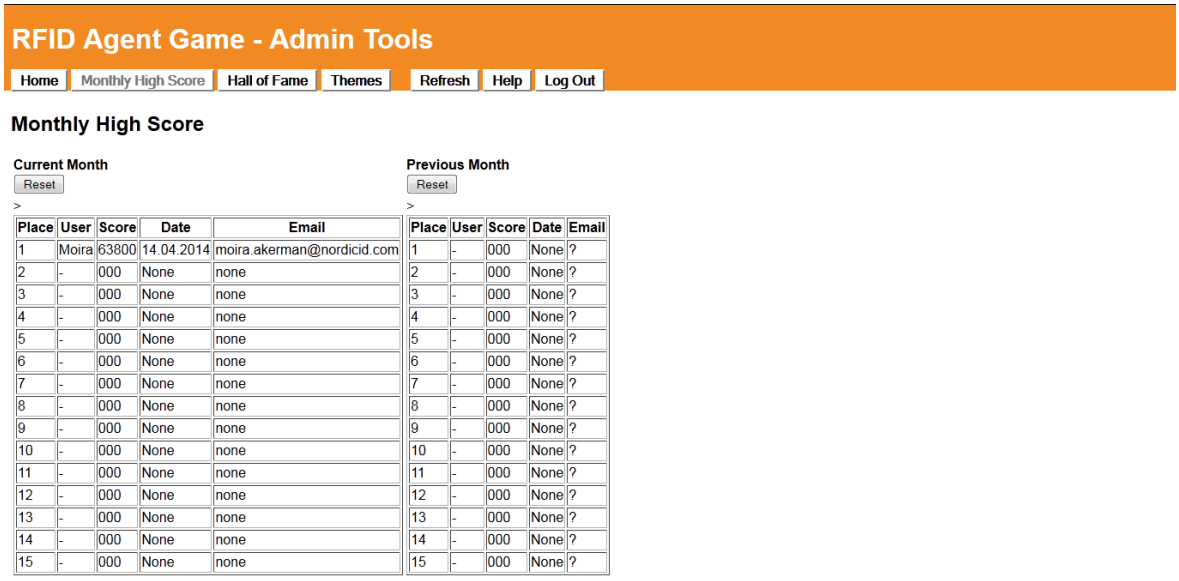
De administrativa verktygen underlättar monitorering av spelet och hantering av teman. För att använda verktygen krävs inloggning. Dessutom är det möjligt att specificera en IP-rymd som begränsar möjligheten att använda verktygen till endast ett nätverk. Mer om användningen av verktygen står det i kap 2.2 Tekniskt i detta arbete.

Administrative tools består av en meny. Klickar man på knapparna, ändras innehållet på sidan med JavaScript.

Figur 3, 4 och 5 visar de mest intressanta sidorna i de administrativa verktygen.



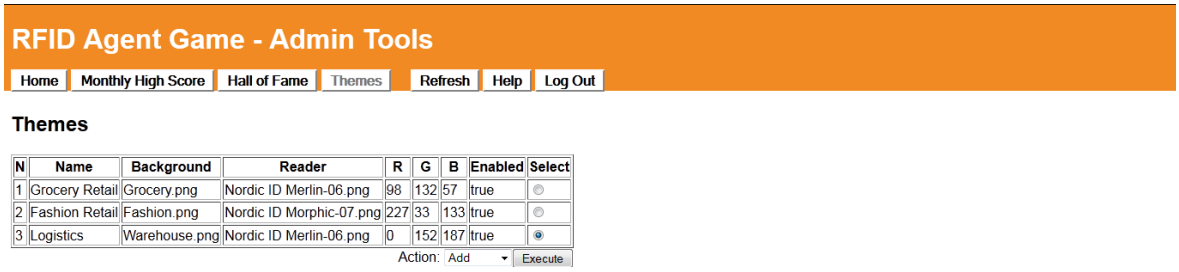
Figur 3. Administrativa verktygens startsida.



Place	User	Score	Date	Email
1	Moira	63800	14.04.2014	moira.akerman@nordicid.com
2	-	000	None	none
3	-	000	None	none
4	-	000	None	none
5	-	000	None	none
6	-	000	None	none
7	-	000	None	none
8	-	000	None	none
9	-	000	None	none
10	-	000	None	none
11	-	000	None	none
12	-	000	None	none
13	-	000	None	none
14	-	000	None	none
15	-	000	None	none

Place	User	Score	Date	Email
1	-	000	None	?
2	-	000	None	?
3	-	000	None	?
4	-	000	None	?
5	-	000	None	?
6	-	000	None	?
7	-	000	None	?
8	-	000	None	?
9	-	000	None	?
10	-	000	None	?
11	-	000	None	?
12	-	000	None	?
13	-	000	None	?
14	-	000	None	?
15	-	000	None	?

Figur 4. Administrativa verktygens månadspoäng.



N	Name	Background	Reader	R	G	B	Enabled	Select
1	Grocery Retail	Grocery.png	Nordic ID Merlin-06.png	98	132	57	true	
2	Fashion Retail	Fashion.png	Nordic ID Morphic-07.png	227	33	133	true	
3	Logistics	Warehouse.png	Nordic ID Merlin-06.png	0	152	187	true	

Figur 5. Administrativa verktygens tema-sektion.

Då ett tema editeras eller läggs till, fyller man i ett formulär, där man kan ladda upp eller använda färdigt uppladdade bilder och påverka de övriga attributen för ett tema. I formuläret finns även förhandsvisning på färdigt uppladdade bilder, samt den unika bubbelfärgen. Färgen demonstreras med en bubbla på vit bakgrund, precis som det kommer att vara i spelet.

5 Testning, problem och lösningar

Testningen skedde på två sätt: i samband med utvecklingen och ur beställarens och användarens synvinkel. Nordic ID deltog i testningen och gjorde kvalitetssäkring. Deras programmerare hade tillgång till koden som syns i webbläsaren, men annars var spelet för dem en Black box, alltså ett program som testas utan vetskap om källkoden, medan det i samband med utvecklingen snarare var en White box, då källkoden är tillgänglig vid

testandet (Eriksson, U. 2008). I testningen användes både systematik och *ad hoc*. Som testmiljö användes olika webbläsare och även några mobila plattformar såsom iPad och Lumia. Testresultaten hade stor påverkan på utvecklingen av spelet, eftersom det ofta skedde parallellt med produktionen. Förutom testning av spelets funktionalitet, testades i början även hur arrayer behandlar objekt, och det visade sig att objekten egentligen lagras som referenser. Detta innebär att ett och samma objekt kan användas via olika arrays. Dessutom framkom att om en referens lagras vidare från en array till en annan, och tas sedan bort från den ursprungliga, går objektet fortfarande att använda via den senare referensen, istället för att hela objektet skulle försvinna.

Under produktionen stötte jag på flera problem. Oftast gick de snabbt att lösa, men några skilde sig från mängden.

5.1 Scrollbars döljs inte i Opera

Internetläsaren Opera kunde visa scrollbars, även då elementet ligger i ett osynligt element. Detta löstes med en CSS-klass som jag döpte till ”scrollable”. Alla element som kan få en scrollbar kan då manipuleras med JavaScript, och det som tar bort scrollbars är att ändra värdet för attributet *overflow* till ”hidden”. *Programlistning 34* visar verktygsfunktionerna som döljer och visar scrollbars.

Listning 34. Programkod i JavaScript som döljer och visar scrollbars i rfid_game_plugin.js.

```
// Forcing scrollbars to be hidden in Opera
function hideSBars() {
      var elems = document.getElementsByClassName('scrollable'), i
= elems.length;
    while(i--)
        elems[i].style.overflow = "hidden";
}
function showSBars() {
      var elems = document.getElementsByClassName('scrollable'), i
= elems.length;
    while(i--)
        elems[i].style.overflow = "auto";
}
```

Att bara dölja scrollbaren var inte tillräckligt. Även om de är dolda, är de fortfarande i vägen för andra scrollbars, så att de inte går att använda. Detta löstes med CSS-attributet z-index. Som utgångspunkt är z-index 0 för alla element, men för alla element som innehåller scrollbar, ändras z-index till 1 när elementet ska visas, och 0 då det döljs med funktionen backToMenu.

Här har vi ett exempel på användningen av z-index med JavaScript, för att lägga credits framför de andra elementen. Koden finns i funktionen showCredits i rfid_game_plugin.js.

```
credits.style.zIndex=1;
```

5.2 Laddning av musik misslyckas

Ett av de större problemen var att ladda musiken. Webbläsare kunde plötsligt bara låta bli att ladda en musikfil helt. Cache kunde lösa detta problem, men även då kunde det i somliga webbläsare, såsom Google Chrome, uppkomma oväntade resultat. De metoder som fanns på Internet för precache fungerade inte i detta fall. Ett *en-efter-en*-system fungerade inte heller. Då jag sökte en lösning, hittade jag howler.js, som gör audioprogrammeringen lättare. Den verkade inte ha en direkt lösning till laddningen av flera olika filer. Andra bra bibliotek är SoundJS, och PreloadJS men de är lite långsamma, och onödigt stora. Jag kom till slutsatsen att det går lättast att låta teknologin utvecklas, så att webbläsarna kan hantera HTML5-audio mer pålitligt.

5.3 Safari animerar långsamt

Det värsta problemet jag inte kunde lösa var att Internetläsaren Safari spelar animationerna mycket långsamt. Detta är ett allmänt problem, och flera webbutvecklare försöker inte ens anpassa HTML5 för bara Safaris skull, utan väntar istället på att Safari utvecklas, såsom utvecklarna på *Code* (Code Computerlove u.å.) gjort.

Senare hittade jag en funktion, *window.requestAnimationFrame(gameLoop)*, som spelar animationerna så snabbt som webbläsaren klarar av. Det är knappast till stor hjälp, men kanske Safari fungerar bättre med den. Jag började inte implementera koden för att den existerande koden är alltför beroende av en inställd frekvens, så requestAnimationFrame kunde medföra mycket extra kodningsarbete.

6 Slutdiskussion

Utvecklingen av RFID Agent Game var en stimulerande och lärorik utmaning. Jag påbörjade examensarbetet ett år tidigare än normalt. Arbetet skedde parallellt med tredje årets studier, så arbetstakten blev konsekvent långsammare än normalt. Beställaren hade ingen tidsgräns, men själv strävade jag att få allt färdigt till våren för att kunna bli utexaminerad till sommaren. Min plan var att få spelet färdigt till årsskiftet, men då var endast basfunktionaliteten klar. Beställaren hade ännu krav på utseendet och en del av funktionaliteten behövde även finslipas. När den skriftliga delen av examensarbetet blev klart, fanns det ännu lite kvar att göra på spelet. Beställaren har varit nöjd med spelets demoversioner, och den slutliga versionen kommer knappast att vara ett undantag.

Till slut vill jag tacka alla som medverkat eller stött mig under arbetet.

7 Sammanfattning

HTML5-spelet RFID Agent Game är en bubbleshooter med beställarens, Nordic IDs tema. I spelet finns även teman som influerats av platser där Nordic Ids produkter utnyttjas, såsom i klädaffärer, matbutiker och lager. Produkterna, RFID-läsarna, finns också med i spelet; bubblorna skjuts iväg från dem.

Bakom spelupplevelsen finns över 3000 rader med kod och ca 27 MB med grafik och ljud. Tillsammans blir det ca 28 MB med data, vilket motsvarar spelets storlek och komplexitet. I produkten ingår även ett administrativt verktyg.

Små problem uppstod med HTML5, eftersom teknologin ännu är under utveckling. Då detta skrevs, behövde spelet ännu finslipas, men det som snart kommer att vara slutresultatet ser lovande ut.

Källförteckning

Code Computerlove (u.å). *Flash vs HTML5*. Webbsida. http://labs.codecomputerlove.com/FlashVsHtml5/ (hämtat 30.1.2014).

dsportes (2011). *IE9 does not play mp3 in audio tag*. Internet Explorer Dev Center. http://social.msdn.microsoft.com/Forums/ie/en-US/b4f58d95-ac27-4a28-a4ae-86477ddfc74f/ie9-does-not-play-mp3-in-audio-tag?forum=iewebdevelopment (hämtat 22.1.2014).

Eriksson, U. (2008). *Test och kvalitetssäkring av IT-system. (2. uppl.)* Lund: Studentlitteratur.

Nesbyte (u.å). *List of The Top 5 Modern Web Browsers.* Webbartikel. http://nesbyte.hubpages.com/hub/Top-5-Modern-Web-Browsers (hämtat 30.1.2014).

Page, K. (2013). *The Benefits of HTML5 vs. Adobe Flash*. Blog. http://www.emaze.com/blog/html5-vs-flash/ (hämtat 25.1.2014).

Shah, N. & Ortíz, G. (2013) *HTML5 Enterprise Application Development* Olton: Packt Publishing Ltd.

Stackoverflow (2014). *Programmeringsforum.* www.stackoverflow.com (hämtat 22.1.2014).

Trasviña, F. (2010). *Introduction to Object-Oriented JavaScript*. Artikel. https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript (hämtat 10.2.2014).

W3Schools (2014). *Webbprogrammeringsskolning.* www.w3schools.com (hämtat 22.1.2014).

Bilaga 1

```
// 1. MAIN FUNCTION
// The function that draws the game on the page, inside the element specified.
// Because there is some loading in the background, the function is divided into 2. The other one is called "continueInsertion"
// and is executed when XHR has finished loading
// fixX and fixY are used when the container is not positioned at the left edge, or there is some trouble with global coordination
// especially used for aiming the reader.
function insertRFIDGame(element, autoplayMusic, fixX,fixY) {
	if(autoplayMusic==null)
		autoplayMusic = false;
	container = element;
	container.style.fontFamily = "Avenir, Arial";
	x_fix = fixX==null?0:fixX;
	y_fix = fixY==null?0:fixY;
	// Uri params is used to remember the user preferences after a session reset.
	var uriParams = getURIParams();
	if(uriParams.m===1)
		musicEnabled = true;
	else if(uriParams.m===0)
		musicEnabled = false;
	else
		musicEnabled = autoplayMusic;
	if(uriParams.s==0)
		soundEnabled=false;
	else soundEnabled=true;
	// Get the container size
	w = container.style.width;
	h = container.style.height;
	// Convert the size to numbers
	w = w.substring(0,w.length-2)*1;
	h = h.substring(0,h.length-2)*1;
	// Define the size value, the base of scaling and positioning everything in the game.
	s = (w < h)?w:h;
	// Center the background at the axis that has more space
	x = (w <= h)?0:(w-s)/2;
```

```
	y = (w <= h)?(h-s)/2:0;
	margin = s*9/493;
	//500/(18*2+480*9/493*2/18)*2
	br = osize/(bubblesPerRow*2+margin*2/bubblesPerRow);
	xoffset = br/4;
	yoffset = xoffset;

	amountRows = Math.floor(osize/(br*2*Math.sin(abr))-yoffset);
	grid = new Grid();
	// Create a bubble model that can be used when pre-rendering the bubbles
	b_img = new Image();
	b_img.src = imgr + 'Bubble_model.png';
	tempc = document.createElement("canvas");
	//tempc.width = br*2*overSampling; tempc.height = br*2*overSampling;
	//tempc.style.width = br*2; tempc.style.height = br*2;
	// Get imgage data for bubble model
	tctx = tempc.getContext("2d");
	b_img.onload = function() {
		tctx.drawImage(b_img,0,0,br*2,br*2);
		// pull the entire image into an array of pixel data, the model
		b_imageData = tctx.getImageData(0, 0, br*2, br*2);
		modelsLoaded++;
	}
	var incompatibilities = checkCompatibility();
	if(incompatibilities.length > 0) {
		container.innerHTML = '<h2 style="padding:4px;">Sorry, but your browser does not support all features of RFID Agent Game.<br>'
		+'RFID Game should work in Internet Explorer 9+, Firefox, Chrome, Opera and Safari<br><br>'
		+"Your browser does not seem to support:<br>"+incompatibilities+"</h2>";
		return false;
	}
	// Inserting the progress bar and writing Progress 1 of 3
	container.innerHTML='<div id="loading" style="position:absolute; left:'+(x+8+s/4)+'px; top:'+(y+8+s/4)+'px;'
	+'width:'+s/2+'px; height:'+s/4+'px;z-index:1;background-color:#FFF;padding:5px;text-align:center;"></div>';
```

```
	loading = document.getElementById("loading");
	loading.innerHTML = '<b style="font-size:'+16/360/1*s+'px;">Loading, please wait</b><br/>Setting Up and Loading Themes<br/>'
	+'<div style="width:100%;height:4%;background-color:#BBB;"><div style="width:33%;height:100%;background-color:#000;"></div></div>';
	// Continue insertion when the themes and models have been loaded
	loadThemes('rfid_game_plugin/themes.xml');
	return true;
}
function continueInsertion() {
	// If themes and models would not be loaded yet, we wait
	//alert(modelsLoaded);
	if(themes==null || modelsLoaded<amountModels) {
		setTimeout(continueInsertion,500);
		return;
	}
	// Write Progress 2
	loading.innerHTML='<b style="font-size:'+16/360/1*s+'px;">Loading, please wait</b>Loading Resources<br/><div style="width:100%;height:2%;background-color:#BBB;">'
	+'<div style="width:66%;background-color:#000;"></div></div>';
	// Remove the loading div because it will be recreated (causes trouble otherwise)
	loading.parentNode.removeChild(loading);
	// Background music
	/*
	mainTheme.loop = true;
	mainTheme.play();
	*/
	/**/
	var tracksToLoad = 5;
	var mainTheme = audioDOM("mainTheme","music/main_theme");
	//alert(tracks[0]);
	mainTheme.addEventListener("error",function() {
		alert("error loading Main Theme");
	});
	mainTheme.addEventListener("canplay",function(){
```

```
          // This track is an example of how little data is
required if the music loops once in its entirety and it has been
cut perfectly.
          // if WebBrowsers and W3C improve the audio.loop, it
will actually do the same thing.
          // using the push method is not adviced, as the
indexing can become random in e.g. Chrome
          tracks[0]=new Track("Main Theme", mainTheme, 0.7, 1,
3.1);

     tracks[0].music[1].addEventListener("canplay",function(){
               if(musicEnabled)mPlayer.Play();

     this.removeEventListener('canplay',arguments.callee,false);
          });

     this.removeEventListener('canplay',arguments.callee,false);
     });
     var m1 = audioDOM("m1","music/The Agent");
     forceLoad(m1);
     m1.addEventListener("error",function() {
          alert("error loading The Agent");
     });
     m1.addEventListener("canplay",function(){
          // This is an example of the simplest way to create a
track. It plays just once with 100 % volume
          tracks[1] = new Track("The Agent",m1);

     this.removeEventListener('canplay',arguments.callee,false);
     });
     var m2 = audioDOM("m2","music/In Action");
     forceLoad(m2);
     m2.addEventListener("error",function() {
          alert("error loading In Action");
     });
     m2.addEventListener("canplay",function(){
          tracks[2] = new Track("In Action",m2, 1, 1, 3.13);

     this.removeEventListener('canplay',arguments.callee,false);
     });
     var m3 = audioDOM("m3","music/Fashion");
     forceLoad(m3);
     mainTheme.addEventListener("error",function() {
```

```
		alert("error loading Fashion");
	});
	m3.addEventListener("canplay",function(){
		tracks[3] = new Track("Fashion", m3, 1, 1, 1, 14.76);

	this.removeEventListener('canplay',arguments.callee,false);
	});
	var m4 = audioDOM("m4","music/Anxiety");
	forceLoad(m4);
	m4.addEventListener("error",function() {
		alert("error loading Anxiety");
	});
	m4.addEventListener("canplay",function(){
		tracks[4] = new Track("Anxiety", m4, 1, 1, 0.5, 14.76);

	this.removeEventListener('canplay',arguments.callee,false);
	});
	checkMusic(tracksToLoad,0);
	/* Example how to save time while testing different values on
loops. (make sure the track is the only one played at the
beginning)
	var mx = audioDOM("m4","music/Anxiety");
	mx.addEventListener("canplay",function(){
		tracks[x]=new Track("Anxiety", m4, 1, 1, 0.5, 14.76);
		// Code that saves time while testing different values,
if the music is longer than 120 seconds, this will save you 120
seconds
		var skip = 120;
		tracks[x].music[0].currentTime = skip;
		tracks[x].loopEnd -=skip;

	tracks[x].music[1].addEventListener("canplay",function(){
			mPlayer.Play(tracks[x]);
		});
	});
	*/


	// Sound effects
	sfx['pop'] = audioDOM("pop_sfx","sfx/pop1");
	sfx['erel'] = audioDOM("erel_sfx","sfx/erel");
	sfx['gameover'] = audioDOM("gmovr","sfx/gameover");
```

```
	// Setting background
	container.style.backgroundRepeat = "no-repeat";
	container.style.backgroundSize = s+"px "+ s+"px";
	container.style.backgroundPosition =  x +"px "+ y +"px";
	// Final canvas size (it will be oversampled to preserve great quality on graphics)
	cs = s*3/4+margin*2;
	// Canvas position
	cx = x+s/8-margin;
	cy = y+s/8-margin;
	// Create the canvas element centered and sized according to the container, with containers for main menu and pre-rendered stuff.
	container.innerHTML += '<canvas id="gameArea" width="'+cs*overSampling+'" height="'+cs*overSampling+'"'
	+'style="width:'+cs+'px; height:'+cs+'px; position:relative;left:'+cx+'px;top:'+cy+'px; border:1px solid #000000;'
	+'background-image:url('+imgr+'Game_space.png);background-repeat:no-repeat;background-size:'+cs+'px '+cs+'px;"></canvas>'
	+'<div id="mainMenu" style="font-size:'+14/360*s+'px;position:absolute; left:'+(x+8)+'px; top:'+(y+8)+'px;'
	+'width:'+s+'px; height:'+s+'px; background-color:#AAAAAA;"></div>'
	+'<div id="menuDisabler" style="position:absolute; left:'+(x+8)+'px; top:'+(y+8)+'px;width:'+s+'px; height:'+s+'px;'
	+'visibility:hidden;"></div>'
	+'<div id="credits" style="position:absolute; left:'+(x+8+s*1/8)+'px; top:'+(y+6+s*1/8)+'px;'
	+'width:'+(s*3.05/4)+'px; height:'+(s*3.05/4)+'px; background-color:#FFFFFF;visibility:hidden;"></div>'
	+'<div id="info" style="position:absolute; left:'+(x+8+s*1/8)+'px; top:'+(y+6+s*1/8)+'px;'
	+'width:'+s*3.05/4+'px; height:'+s*3.05/4+'px; background-color:#FFFFFF;visibility:hidden;"></div>'
	+'<div id="highscore" style="position:absolute; left:'+(x+8+s*1/8)+'px; top:'+(y+6+s*1/8)+'px;'
	+'width:'+s*3.05/4+'px; height:'+s*3.05/4+'px; background-color:#FFFFFF;visibility:hidden;"></div>'
	+'<div id="options" style="position:absolute; left:'+(x+8+s*1/8)+'px; top:'+(y+8+s*1/8)+'px;'
	+'width:'+s*3.05/4+'px; height:'+s*2.7/4+'px; background-color:#FFFFFF;visibility:hidden;"></div>'
	+'<div id="gameOver" style="position:absolute; left:'+(x+8+s/4)+'px; top:'+(y+8+s/4)+'px;'
```

```
      +'width:'+s/2+'px;                              height:'+s/4+'px;
visibility:hidden;"></div>'

      +'<div id="prerenderer" style="visibility:hidden;"></div>'

      +'<div          id="loading"          style="position:absolute;
left:'+(x+8+s/4)+'px; top:'+(y+8+s/4)+'px;'

      +'width:'+s/2+'px;     height:'+s/4+'px;z-index:1;background-
color:#FFF;text-align:center;padding:5px;"></div>';


      // DOMs

      // Reload the loading DOM, as innerHTML changes can mess up
things

      loading = document.getElementById("loading");

      // Rewrite Progress 2

      loading.innerHTML='<b                                 style="font-
size:'+16/360/1*s+'px;">Loading,    please    wait</b><br/>Loading
Resources<br/><div          style="width:100%;height:4%;background-
color:#BBB;">'

      +'<div                style="width:66%;height:100%;background-
color:#000;"></div></div>';


      gameArea = document.getElementById("gameArea");

      mainMenu = document.getElementById("mainMenu");

      menuDisabler = document.getElementById("menuDisabler");

      credits = document.getElementById("credits");

      info = document.getElementById("info");

      highscore = document.getElementById("highscore");

      options_div = document.getElementById("options");

      gameOver_div = document.getElementById("gameOver");

      renderer = document.getElementById("prerenderer");


      // Setup main menu

      /*

      mainMenu.style.backgroundImage                                    =
"url('"+imgr+"bg/main_menu.png')";

      mainMenu.style.backgroundRepeat = "no-repeat";

      mainMenu.style.backgroundSize = s+"px "+ s+"px";

      */

      var themeOptionsHTML = "";

      for(var i=0;i<themes.length;i++) {

            if(themes[i].tenabled=="1")

                  themeOptionsHTML            +=            '<option
value="'+i+'">'+themes[i].name+'</option>';

      }
```

```
	// Due to an annoying code coloring bug in Adobe Dreamweaver CS6, some mathemathical expressions includes "/1"


	// Building up the main menu

	mainMenu.innerHTML =

	// Top

	'<div style="width:'+s+'px;height:'+100*s/932+'px;'

	+'background-image:url('+ebgr+'Button-100-x-932.png);background-size:'+s+'px '+100*s/932+'px;background-repeat:no-repeat;">'

	+'<img src="'+ebgr+'Choose-vertical-theme.png" width="43%"'

	+'style="position:absolute;left:'+s/45+'px;top:'+25*s/932+'px;top:'+25*s/932/1+'px;"/>'

	//+'<select id="theme_sel" onchange="changeTheme();" style="width:'+s*0.45+'px; height:'+50*s/932+'px; font-size:'+14/360*s+'px;'

	//+'position:absolute; top:'+25*s/932+'px; right:'+s/45+'px;">'+themeOptionsHTML+'</select>'

	+'<div id="theme_sel" style="width:'+s*0.45+'px; height:'+50*s/932+'px; font-size:'+14/360*s+'px;'

	+'position:absolute; top:'+16*s/932+'px; right:'+s/45+'px;"></div></div>'

	// Middle

	+'<img id="themePreview" width="'+s+'" height="'+s*632/932/1+'" src="'+themes[0].bg.src+'"/>'

	+'<img src="'+ebgr+'Play-button.png" onclick="startGame();" style="cursor:pointer;'

	+'position:absolute;left:'+(s/3)+'px;top:'+(25*s/932+s*632/(932*2*3.5)+s/6)+'px;width:'+s/3+'px;height:'+s/3+'px;"/>'

	// Bottom

	+'<img src="'+ebgr+'Highest-score-button.png" width="'+s*0.51+'" height="'+s*0.11+'"'

	+'style="font-size:'+16/320/1*s+'px;position:absolute;right:'+0+'px;bottom:'+s*0.11+'px;cursor:pointer;" onclick="showHighScore();"/>'

	+'<img src="'+ebgr+'Instructions-button.png" width="'+s*0.51+'" height="'+s*0.11+'"'

	+'style="font-size:'+16/320/1*s+'px;position:absolute;left:'+0+'px;bottom:'+s*0.11+'px;cursor:pointer;" onclick="showInstructions();"/>'

	+'<img src="'+ebgr+'Credits-button.png" width="'+s*0.51+'" height="'+s*0.11+'"'

	+'style="font-size:'+16/320/1*s+'px;position:absolute;right:'+0+'px;bottom:'+0+'px;cursor:pointer;" onclick="showCredits();"/>'
```

```
      +'<img   src="'+ebgr+'Options-button.png"   width="'+s*0.51+'"
height="'+s*0.11+'"'
      +'style="font-
size:'+16/320/1*s+'px;position:absolute;left:'+0+'px;bottom:'+0+'p
x;cursor:pointer;" onclick="showOptions();"/>';


      theme_sel                          =                          new
DropDownMenu(document.getElementById("theme_sel"),"theme_sel",ddke
ys,ddonhover);
      theme_sel.onchange = changeTheme;
      //theme_sel = document.getElementById("theme_sel");
      themePreview = document.getElementById("themePreview");


      // Setup high score
      /*
      highscore.style.backgroundImage                                  =
"url('"+imgr+"bg/credits.png')";
      highscore.style.backgroundRepeat = "no-repeat";
      highscore.style.backgroundSize = s+"px "+s+"px";
      */


      highscore.innerHTML='<button          onclick="backToMenu();"
style="position:absolute;top:0px;left:0px;">Back                  to
Menu</button>';


      // Setup instructions
      info.innerHTML='<div                          style="background-
color:#EFEBEA;height:'+s*0.1+'px;
width:100%;position:absolute;top:0px;right:0px;">'
      +'<div                         style="text-align:center;font-
size:24px;position:relative;top:25%;">INSTRUCTIONS</div>'
      +'<img     width="'+s*0.16+'"     src="'+ebgr+'back-text.png"
onclick="backToMenu();"'
      +'style="position:absolute;top:23%;right:1%;cursor:pointer;"/
></div>'


      +'<div                                    class="scrollable"
style="overflow:auto;padding:10%;width:80.1%;height:67%;position:a
bsolute;top:'+s*0.1+'px;">'
      +'<b style="font-size:18px;">Aim of the game:</b><ul><li>Scan
as many bubbles as possible.</li>'
      +'<li>Release as many orange elements as possible.</li>'
      +'<li>Score as many points as possible and try to keep your
energy level high.</li></ul>'
```

```
      +'<img          src="'+imgr+'Instructions-pic-of-bubbles-and-
elements.png" width="90%" style="max-width:280px;" /><br/><br/>'

      +'<b style="font-size:18px;">How to play:</b><ul><li>Aim at
and scan bubbles in the game area. '

      +'The bubble bar at the left bottom of the game will show you
what color the next five bubbles have.</li>'

      +'<li>The bubbles are scanned when you hit a cluster of three
or more same-colored bubbles.</li>'

      +'<li>Release elements by scanning the bubbles below them so
that the elements drop to the bottom.</li>'

      +'<li>Scanned bubbles grant you points and energy. Released
elements grant you even more.</li>'

      +'<li>Do not let the grid of bubbles and elements reach the
bottom of the game area. '

      +'New bubbles and elements will fill the game field at an
increasing rate during the game. '

      +'New elements will appear only if you have cleared 3/4th of
the rows. They appear more and more seldom as you advance in the
game.</li>'

      +'<li>Avoid running out of energy. The energy bar at the
right bottom of the game shows you how much energy you have
left.</li></ul>'

      +'<b style="font-size:18px;">Controls:</b><ul><li>Using the
mouse: Move the mouse to aim and scan bubbles by pressing the left
mouse button</li>'

      +'<li>Using the keyboard: Use the arrow keys LEFT and RIGHT
or the number keys 4 and 6 to aim. '

      +'Scan bubbles by pressing the UP arrow key or number key
8.</li></ul>'

      +'<b style="font-size:18px;">Hints:</b><ul><li>Try to scan
enough bubbles to cut off an entire section '

      +'- causing all the bubbles and elements underneath to be
scanned and released as well.</li>'

      +'<li>Collect new big clusters of bubbles. The more you
manage to scan at a time, the higher the points.</li>'

      +'<li>You can use the right and left walls to bounce the
bubbles to their desired location.</li></ul>'

      +'</div>';


      // Setup Options

      options_div.innerHTML     =     '<div     style="background-
color:#EFEBEA;height:'+s*0.1+'px;
width:100%;position:absolute;top:0px;right:0px;">'

      +'<div                          style="text-align:center;font-
size:24px;position:relative;top:25%;">OPTIONS</div>'

      +'<img     width="'+s*0.16+'"     src="'+ebgr+'back-text.png"
onclick="optionsCheck();"'
```

```
      +'style="position:absolute;top:23%;right:1%;cursor:pointer;"/
></div>'

      // Music and Sound

      +'<div      style="width:100%;      height:'+(s*1.31)/2+'px;
position:relative;top:'+s*0.1+'px;background-repeat:no-repeat;'

      +'background-image:url('+ebgr+'Options_white-field.png);
background-size:100% 100%;font-size:'+12/320/1*s+'px;">'

      +'<table                                              width="90%"
style="position:relative;top:16%;left:10%;">'

      +'<tr><td style="width:40%;">Music</td>'

      +'<td    style="width:30%;">On     <img    id="music_cb1"
width="16%"                                 src="'+ebgr+'checkbox.png"
onclick="playMuteMusic(1);"/></td>'

      +'<td    style="width:30%;">Off     <img    id="music_cb2"
width="16%"                                 src="'+ebgr+'checkbox.png"
onclick="playMuteMusic(0);"/></td>'

      +'</tr><tr><td style="width:40%;">Sound FX</td>'

      +'<td    style="width:30%;">On     <img    id="sound_cb1"
width="16%"                                 src="'+ebgr+'checkbox.png"
onclick="playMuteSFX(1);"/></td>'

      +'<td    style="width:30%;">Off     <img    id="sound_cb2"
width="16%"                                 src="'+ebgr+'checkbox.png"
onclick="playMuteSFX(0);"/></td></tr>'

      +'</table></div>'

      // Game Size

      +'<div    style="width:100%;    height:'+(s*3/4-s*0.1)/2+'px;
position:absolute;top:'+(s*0.1+(s*3/4-s*0.1)/2)+'px;">'

      +'<table                                              width="90%"
style="position:relative;top:25%;left:10%;" title="The game will
reload in the chosen scale, when &amp;quot;BACK&amp;quot; '

      +'is clicked. The game can not become smaller than 480x480
pixels or larger than 1200x1200 pixels.">'

      +'<tr><td  style="width:40%;font-size:'+10/320/1*s+'px;">Game
Size</td>'

      +'<td   title="Scales   the   game   for   Maximised   mode"
style="width:40%;font-size:'+10/320/1*s+'px;">Full Window</td>'

      +'<td    style="width:20%;font-size:'+10/320/1*s+'px;"><input
id="size_cb1" type="checkbox" onclick="setSize(1);"/></td>'

      +'</tr><tr><td                              style="width:40%;font-
size:'+10/320/1*s+'px;"> </td>'

      +'<td title="Scales the game for Full Screen mode (F11)"
style="width:40%;font-size:'+10/320/1*s+'px;">Full Screen</td>'

      +'<td    style="width:20%;font-size:'+10/320/1*s+'px;"><input
id="size_cb2" type="checkbox" onclick="setSize(2);"/></td>'

      +'</tr><tr><td                              style="width:40%;font-
size:'+10/320/1*s+'px;"> </td>'
```

```
      +'<td title="Scales the game to fit in the current window"
style="width:40%;font-size:'+10/320/1*s+'px;">Current Window</td>'

      +'<td    style="width:20%;font-size:'+10/320/1*s+'px;"><input
id="size_cb3" type="checkbox" onclick="setSize(3);"/></td></tr>'

      +'</tr><tr><td                          style="width:40%;font-
size:'+10/320/1*s+'px;"> </td>'

      +'<td title="Choose this if you do not want to change the
size"             style="width:40%;font-size:'+10/320/1*s+'px;">No
Change</td>'

      +'<td    style="width:20%;font-size:'+10/320/1*s+'px;"><input
id="size_cb4"          type="checkbox"          checked="checked"
onclick="setSize(4);"/></td>'

      +'</tr></table></div>';


      setTimeout(function(){playMuteMusic(musicEnabled?1:0);},2000)
;

      setTimeout(function(){playMuteSFX(soundEnabled?1:0);},2000);


      // Setup credits

      /*

      credits.style.backgroundImage                                =
"url('"+imgr+"bg/credits.png')";

      credits.style.backgroundRepeat = "no-repeat";

      credits.style.backgroundSize = s+"px "+s+"px";

      */


      credits.innerHTML='<div                    style="background-
color:#EFEBEA;height:'+s*0.1+'px;
width:100%;position:absolute;top:0px;right:0px;">'

      +'<div                          style="text-align:center;font-
size:24px;position:relative;top:25%;">CREDITS</div>'

      +'<img     width="'+s*0.16+'"     src="'+ebgr+'back-text.png"
onclick="backToMenu();"'

      +'style="position:absolute;top:23%;right:1%;cursor:pointer;"/
></div>'


      +'<div                                  class="scrollable"
style="overflow:auto;width:100%;height:87%;position:absolute;top:'
+s*0.1+'px;text-align:center;">'

      +'<h2>Creators</h2>Hanna                &Ouml;stman<br/>Markus
Silvennoinen<br/>Mirva Saarij&auml;rvi'

      +'<h2>Graphics                            Artist</h2>Hanna
&Ouml;stman<h2>Programming</h2>Markus
Silvennoinen<h2>Music</h2>Markus Silvennoinen'
```

```
      +'<h2>SFX</h2>Markus                    Silvennoinen<h2>Quality
Assurance</h2>Hanna  &Ouml;stman<br/>Kirsikka  Dr&auml;ger<br/>Pia
Jokinen'
      +'<h2>Thanks to</h2>Viktor Candolin<br/><br/></br>'
      +'<img src="'+imgr+'logo.png"/><br/><br/><br/><br/></div>';


      // Setup Game Over
      gameOver_div.innerHTML  =  '<img  src="'+ebgr+'Game-over.png"
width="100%"/>';


      //  Pre-render  bubbles  into  storage,  note  the  order  of
rendering determines the order in the bubbles-array
      // Black
      renderBubble(0,0,0);
      // Dark Grey
      renderBubble(152,152,152);
      // Light Grey
      renderBubble(220,219,219);


      ctx=gameArea.getContext("2d");


}
```

Bilaga 2

```
// function that returns all combinations from a sertain point
// Algorithm: Find and list all combinations that have not been
listed yet around the current bubble, and perform the same task
// on the found ones
function getCombinations(si,sj) {
  //si+" "+sj);
  if(!shooting) resetGame("No Cheating!");
  var combs = new Array(1);
  combs[0] = {i:si,j:sj};
  var ix=0;
  var ti,tj,i,j,listed;
  while(combs.length>ix) {
  i = combs[ix].i;
  j = combs[ix].j;
  // Search new combinations.
  // Check the cells in front, if there are any
  if(i>0) {
    // Check front left
    if(j>0 || (j==0&&i%2==1)) {
      ti = i-1;
      tj = j-1+i%2;
      if(grid.cells[ti][tj].v!=null) {
        if(grid.cells[ti][tj].v.type=="bubble") {

if(grid.cells[ti][tj].v.colorType==grid.cells[i][j].v.colorType) {
          // Check if the bubble is already in the list
          listed = false
          for(var jx=0;jx<combs.length;jx++) {
            if(combs[jx].i==ti&&combs[jx].j==tj)
                listed = true;
           }
           if(!listed)
            combs.push({i:ti,j:tj});
         }
       }
     }
   }
// Front right
if(j<bubblesPerRow-1 || (j==bubblesPerRow-1&&i%2==0)) {
```

```
 ti = i-1;
 tj = j+i%2;
 if(tj<bubblesPerRow) {
   if(grid.cells[ti][tj].v!=null) {
     if(grid.cells[ti][tj].v.type=="bubble") {

if(grid.cells[ti][tj].v.colorType==grid.cells[i][j].v.colorType) {
       // Check if the bubble is already in the list
       listed = false
        for(var jx=0;jx<combs.length;jx++) {
          if(combs[jx].i==ti&&combs[jx].j==tj)
            listed = true;
          }
          if(!listed)
            combs.push({i:ti,j:tj});
        }
      }
     }
    }
  }
  // Check middle
  // Middle left
  if(j>0) {
    ti = i;
    tj = j-1;
    if(grid.cells[ti][tj].v!=null) {
      if(grid.cells[ti][tj].v.type=="bubble") {

if(grid.cells[ti][tj].v.colorType==grid.cells[i][j].v.colorType) {
     // Check if the bubble is already in the list
     listed = false
     for(var jx=0;jx<combs.length;jx++) {
        if(combs[jx].i==ti&&combs[jx].j==tj)
          listed = true;
      }
      if(!listed)
        combs.push({i:ti,j:tj});
     }
   }
```

```
   }
  }
  // Middle right
  if(j<bubblesPerRow-1) {
    ti = i;
    tj = j+1;
    if(grid.cells[ti][tj].v!=null) {
      if(grid.cells[ti][tj].v.type=="bubble") {

if(grid.cells[ti][tj].v.colorType==grid.cells[i][j].v.colorType) {
        // Check if the bubble is already in the list
        listed = false
        for(var jx=0;jx<combs.length;jx++) {
          if(combs[jx].i==ti&&combs[jx].j==tj)
            listed = true;
        }
        if(!listed)
          combs.push({i:ti,j:tj});
      }
    }
   }
  }
  // Check behind
  if(i<amountRows-1) {
    // Check behind left
    if(j>0 || (j==0&&i%2==1)) {
      ti = i+1;
      tj = j-1+i%2;
      if(grid.cells[ti][tj].v!=null) {
        if(grid.cells[ti][tj].v.type=="bubble") {

if(grid.cells[ti][tj].v.colorType==grid.cells[i][j].v.colorType) {
        // Check if the bubble is already in the list
        listed = false
        for(var jx=0;jx<combs.length;jx++) {
          if(combs[jx].i==ti&&combs[jx].j==tj)
            listed = true;
        }
        if(!listed)
          combs.push({i:ti,j:tj});
```

```
          }
        }
      }
    }
    // Behind right
    if(j<bubblesPerRow-1 || (j==bubblesPerRow-1&&i%2==0)) {
      ti = i+1;
      tj = j+i%2;
      if(grid.cells[ti][tj].v!=null) {
        if(grid.cells[ti][tj].v.type=="bubble") {
if(grid.cells[ti][tj].v.colorType==grid.cells[i][j].v.colorType) {
        // Check if the bubble is already in the list
        listed = false
        for(var jx=0;jx<combs.length;jx++) {
          if(combs[jx].i==ti&&combs[jx].j==tj)
            listed = true;
        }
        if(!listed)
          combs.push({i:ti,j:tj});
       }
      }
     }
    }
   }
   ix++;
 }
 return combs;
}
```

Bilaga 3

```
// Used for pre-rendering
function renderBubble(red,green,blue) {
	if(restart==0) {
		renderer.innerHTML += '<canvas id="b'+bindex+'"
width="'+br*2*overSampling+'" height="'+br*2*overSampling
		+'"
style="width:'+br*2+'px;height:'+br*2+'px;"></canvas>';
	}
	else bindex--;
	var b = document.getElementById("b"+bindex);
	var bctx = b.getContext("2d");
	// If game is restarted and the theme is changed, we must use
the previous canvas.
	if(restart==2) {
		bctx.clearRect(0,0,b.width,b.height);
	}
	//bctx.drawImage(b_img,0,0);

	// Get the model image data and adjust the color...
	var customData = b_imageData;
	// ... Pixel-by-pixel
	for (var i=0;i<customData.data.length;i+=4){
		// Change visible color into the desired color
		if(customData.data[i+3]>0){
			// change to your new rgb
			customData.data[i]=red;
			customData.data[i+1]=green;
			customData.data[i+2]=blue;
		}
	}
	// put the altered data on the pre-render canvas
	bctx.putImageData(customData,0,0);

	b_models[bindex]=b;
	bindex++;
}
```

Bilaga 4

```
function loadThemes(url){
      // Browsers that require ActiveXObject do not tend to support
HTML5 canvas, so they can be ignored here
      xmlhttp.onreadystatechange = getThemes; // Defined next
      xmlhttp.open("POST",url,false);
      xmlhttp.send();
}
// This is an event handler, but it is related to the loadThemes -
function
function getThemes(evtXHR) {
      if (xmlhttp.readyState == 4 && xmlhttp.status == 200){
            var a= new Array();
            ddkeys = new Array();
            // Trying to get nodevalues from a responceXML can be a
real pain in the ass, so we do it manually instead.
            var xmlstr=xmlhttp.responseText;
            // i = Array index, j = theme tag index, ts = childtag
start index, te = childtag end index
            var i = 0;
            var j = xmlstr.indexOf("<theme>")+7;
            var ts = 0;
            var te = 0;
            while(j != -1){
                  a[i] = new Object();
                  // Get name
                  ts = xmlstr.indexOf("<name>", j);
                  te = xmlstr.indexOf("</name>", ts);
                  a[i].name = xmlstr.substring(ts+6,te);
                  // Get background image from url
                  ts = xmlstr.indexOf("<background>", j);
                  te = xmlstr.indexOf("</background>", ts);
                  a[i].bg = new Image();
                  a[i].bg.src =
imgr+"bg/"+xmlstr.substring(ts+12,te);
                  //a[i].bg.onload = countImagesLoaded;
                  // Get reader image  from url
                  ts = xmlstr.indexOf("<reader>", j);
                  te = xmlstr.indexOf("</reader>", ts);
                  a[i].reader = new Image();
```

```
                a[i].reader.src =
imgr+"readers/"+xmlstr.substring(ts+8,te);
                // Dropdown menu images
                ts = xmlstr.indexOf("<ddimage>", j);
                te = xmlstr.indexOf("</ddimage>", ts);
                a[i].ddimage = new Image();
                a[i].ddimage.src =
ddkeys[ddkeys.push(ddmr+xmlstr.substring(ts+9,te))-1];
                ts = xmlstr.indexOf("<ddhover>", j);
                te = xmlstr.indexOf("</ddhover>", ts);
                a[i].ddhover = new Image();
                a[i].ddhover.src = ddmr+xmlstr.substring(ts+9,te);

                // Get RGB-color values
                ts = xmlstr.indexOf("<red>", j);
                te = xmlstr.indexOf("</red>", ts);
                a[i].red = xmlstr.substring(ts+5,te);
                ts = xmlstr.indexOf("<green>", j);
                te = xmlstr.indexOf("</green>", ts);
                a[i].green = xmlstr.substring(ts+7,te);
                ts = xmlstr.indexOf("<blue>", j);
                te = xmlstr.indexOf("</blue>", ts);
                a[i].blue = xmlstr.substring(ts+6,te);
                ts = xmlstr.indexOf("<enabled>", j);
                te = xmlstr.indexOf("</enabled>", ts);
                a[i].tenabled = xmlstr.substring(ts+9,te);
                // Go to next theme if there is one, else we are
done with the loop
                j = xmlstr.indexOf("<theme>", j+7);
                i++;
           }
           themes = a;
           ddonhover = new Array(ddkeys.length);
           for(var i=0;i<ddkeys.length;i++)
                ddonhover[i] =
ddkeys[i].substring(0,ddkeys[i].lastIndexOf('.'))+'-hover-
on'+ddkeys[i].substr(ddkeys[i].lastIndexOf('.'));
           // The insertion should finally be ready to continue
           continueInsertion();
     }
}
```

Bilaga 5

```
function generateContentOnRows(rows) {
	grid.PushContents(rows);
	var content;
	var contentFound;
	var elementPlaced;
	for(var i=0;i<rows;i++) {
		for(var j=0;j<bubblesPerRow;j++) {
			if(grid.cells[i][j].v!=null) continue;
			contentFound = false;
			elementPlaced = false;
			// Check if there is a chance to generate an element and then, with the
			// probability of 1 in bubblesPerRow, an element is generated, but only if it does not overlap existing ones
			if(j<bubblesPerRow-3&&j>0&&i<rows-3) {
				if(randomInt(0,bubblesPerRow)==0) {
					for(var k=1;k<4&&!contentFound;k++) {
						contentFound = grid.cells[i][j+k].v!=null || grid.cells[i+1][j+k-(i+1)%2].v!=null;
					}
					if(!contentFound) {
						// Place element
						elements[el] = randomElement();
						for(var m=0;m<3;m++) {
							grid.cells[i][j+m].v = elements[el-1];
							grid.cells[i+1][j+m-(i+1)%2].v = elements[el-1];
						}
						elementPlaced = true;
					}
				}
			}
			if(!elementPlaced) {
				bubbles[bl] = randomBubble();
				grid.cells[i][j].v = bubbles[bl-1];
			}
			// Set the position of the bubble or element according to its cell
			grid.cells[i][j].v.x = grid.cells[i][j].x;
```

```
                grid.cells[i][j].v.y = grid.cells[i][j].y;
                grid.cells[i][j].v.gi = i;
                grid.cells[i][j].v.gj = j;
            }
        }
    }
```

Bilaga 6

```
function IncomingBubbleLine() {
	this.x = margin*1.2+br/3;
	this.y = osize-margin*2-br/3;
	this.w = 12*br;
	this.h = 7/3*br;
	// Stores the bubbles that are going to be fired
	this.list = new Array(5);
	// Fill the list in the beginning
	for(var i=0;i<5;i++) {
		this.list[i] = randomBubble();
		this.list[i].x = this.x+br/3+(4-i)*(br*7/3);
		this.list[i].y = this.y+br/3;
		bubbles.push(this.list[i]);
	}
}
// This function was intended to be a prototype of the IncomingBubbleLine, but as it is executed inside one of its methods
// it must be defined on the outside, and because there is just one instance, the reference can be direct
function Animate(transitions,current) {
	for(var i=0;i<4;i++) {
		bline.list[i].x += (br*7/3)/transitions;
	}
	if(current>1)
		setTimeout(function(){Animate(transitions,current-1)},1000/fps);
	else {
		// Add a new bubble in the end of the line and set its position
		bline.list.push(randomBubble());
		bline.list[4].x = bline.x+br/3;
		bline.list[4].y = bline.y+br/3;
		bubbles.push(bline.list[4]);
	}
}
// function for releasing a bubble from the list. It will also generate a new bubble at the end of the list
IncomingBubbleLine.prototype.Release = function() {
	// Releases the first bubble in the line and places it in front of the reader, while the line proceeds
```

```
      var b=this.list.shift();
      //reader.bc-br centers the bubble image at the reader's base-
coordinates.
      //The rest positions the bubble at the top edge of the
reader.
      b.x = reader.bx-br-reader.h*Math.cos(reader.a+Math.PI/2);
      b.y = reader.by-br-reader.h*Math.sin(reader.a+Math.PI/2);
      setTimeout(function(){Animate(4,4)},1000/fps);
      return b;
}
```

Bilaga 7

```
// Gridsystem containing bubbles and elements
function Grid() {
	// two-dimensional array
	this.cells = new Array(amountRows);
	var gx, gy;
	for(var i = 0;i<amountRows;i++) {
		this.cells[i] = new Array(bubblesPerRow);
		for(var j = 0; j<bubblesPerRow;j++) {
			// The formula for gx assumes that angle between rows is 60
			//(bubbles on the same line don't have space between each other)
			gx = margin + xoffset + i%2*2*br*Math.cos(abr) + j*2*br;
			gy = margin + yoffset + i*2*br*Math.sin(abr);
			// v can be used as a reference to the object, but it is rather used to check if the cell contains something, a value.
			this.cells[i][j] = {v:null,x:gx,y:gy};
		}
	}
	return this;
}
Grid.prototype.PushContents = function(rows) {
	// Check "game over" by checking the pushed values would exceed the grid limit
	for(var ix=0;ix<bubblesPerRow;ix++) {
		if(this.cells[amountRows-rows][ix].v!=null) {
			gameOver();
			return;
		}
	}
	// Copy values forward and make the "push effect"
	// As elements only store the top left corner's x and y, the scanning direction is optimal from bottomright to topleft
	for(var i=amountRows-1;i>=rows;i--) {
		for(var j=bubblesPerRow-1;j>=0;j--) {
			this.cells[i][j].v = this.cells[i-rows][j].v;
			// Set the new positions of the pushed object, if it is an object, that is.
			if(this.cells[i][j].v!=null) {
```

```
                        this.cells[i][j].v.x = this.cells[i][j].x;
                        this.cells[i][j].v.y = this.cells[i][j].y;
                        this.cells[i][j].v.gi = i;
                        //gj is unchanged
                  }
            }
      }
      // Clear the old duplicates
      for(var i=0;i<rows;i++) {
            for(var j=0;j<bubblesPerRow;j++) {
                  this.cells[i][j].v = null;
            }
      }
}
// Function that returns the y-coordinate for a specified row's
index
Grid.prototype.RowYFromI = function(rowIndex) {
      return margin + yoffset + rowIndex*2*br*Math.sin(abr);
}
// Function that returns row index from a y value
Grid.prototype.RowIFromY = function(y) {
      var ti = Math.floor((y-margin-yoffset)/(2*br*Math.sin(abr)));
      return Math.min(Math.max(0,ti),amountRows-1);
}
// Same two functions as above but for the x-coordinate and column
Grid.prototype.ColumnXFromIJ = function(rowI,columnJ) {
      return margin + xoffset + rowI%2*2*br*Math.cos(abr) +
columnJ*2*br;
}
Grid.prototype.ColumnJFromXI = function(x,rowI) {
      var tj = Math.floor((x-margin-xoffset-
rowI%2*2*br*Math.cos(abr))/(2*br));
      // The returned value must be within the allowed interval
      return Math.min(Math.max(0,tj),bubblesPerRow-1);
}
```

Bilaga 8

```
function startGame() {
	hideSBars();
	var s_check_xhr = new XMLHttpRequest();
	s_check_xhr.onreadystatechange = function() {
		if (s_check_xhr.readyState == 4 && s_check_xhr.status
== 200){
			var resp = s_check_xhr.responseText;
			//alert(resp);
		}
	}
	s_check_xhr.open("POST","rfid_game_plugin/session_controller.
php",true);
	s_check_xhr.setRequestHeader("Content-type","application/x-
www-form-urlencoded");
	var
code=""+v+startGame.toString()+updateScore.toString()+gameLoop.toS
tring()+resetGame.toString()+playAgain.toString()+gameOver.toStrin
g()
	+bubbleHitTest.toString();
	var args =
"status=0&score="+encodeURIComponent(score)+"&pops="+encodeURIComp
onent(bPops)+"&rels="+encodeURIComponent(eRels)+"&energy="+encodeU
RIComponent(energy)+"&code="+encodeURIComponent(code);
	//alert(code);
	s_check_xhr.send(args);


	/* If something has to be done on the serverside when user
exits or refreshes the page.
	window.addEventListener("beforeunload",function() {
		var s_check_xhr2 = new XMLHttpRequest();


	s_check_xhr2.open("POST","rfid_game_plugin/session_controller
.php",true);
		s_check_xhr2.setRequestHeader("Content-
type","application/x-www-form-urlencoded");
		var
code=""+v+startGame.toString()+updateScore.toString()+gameLoop.toS
tring()+resetGame.toString()+gameOver.toString()
		+bubbleHitTest.toString();
		var args =
"status=4&score="+score+"&pops="+bPops+"&rels="+eRels+"&code="+cod
e;
		s_check_xhr2.send(args);
		return null;
```

```
	},false);
	*/
	// Pre-render the theme-specific bubble, set the background
and load the Reader image
	if(restart!=1) {
		if(restart==2) {
			b_models.pop();
		}
		var thm = themes[theme_sel.value];
		renderBubble(thm.red,thm.green,thm.blue);
		reader=new Reader(thm.reader);
		// Draw background
		var src = thm.bg.src;
		src =
src.substring(0,src.lastIndexOf("."))+"G"+src.substr(src.lastIndex
Of("."));
		container.style.backgroundImage = "url(\""+src+"\")";
	}
	// Hide main menu
	mainMenu.style.visibility = "hidden";
	// Generate contents
	generateContentOnRows(Math.round(amountRows*3/5));
	// Create the line of incoming bubbles
	bline = new IncomingBubbleLine();
	// Set the scaling size used during rendering on gameArea-
canvas
	//ctx.scale((s/2)/osize,(s/2)/osize);
	// scaling factor = newSize/originalSize;
	if(restart==0) {
		var s_factor = (s*0.75)/osize*overSampling;
		ctx.scale(s_factor,s_factor);
		// Try scaling with CSS
		/*
		gameArea.style.width *= s_factor;
		gameArea.style.height *= s_factor;
		*/
	}
	// Draw
	renderGame();
	// Draw reader separately, as its appearance changes on mouse
movement
```

```
	ctx.drawImage(reader.image,reader.x,reader.y,reader.w,reader.
h);
	container.addEventListener("mousemove",mouseMove,false);
	container.addEventListener("click",onClick,false);
	document.addEventListener("keydown",onKeyDown,false);
	document.addEventListener("keyup",onKeyUp,false);
	// Add support for touch events too
	container.addEventListener("touchstart",mouseMove,false);
	container.addEventListener("touchmove",mouseMove,false);
	container.addEventListener("touchend",onClick,false);
	// Hide the cursor when it is directly over the gameArea.
	//gameArea.style.cursor="none";
	// Do not let all functionality start at once. Otherwise a
bubble is shot when clicking the Play-button.
	setTimeout(applyStart,20);
	setTimeout(updateScore,5000);
}
function applyStart() {
	gameOn = true;
	gameLoop();
}
```

Bilaga 9

```
// This function acts like the "EnterFrame"-event handler in Flash
function gameLoop() {
	pushCount += 1.0/fps;
	time += 1.0/fps;
	energy-= 1.0/fps;
	eMeter.energy = energy;
	if(energy<=0 && gameOn) {
		gameOver();
	}
	// The base interval (in seconds) for adding more content
	var pushRate = 90;
	// Difficulty is basically dependent on time and score
	// New content will be placed 1 second faster for every "700" in score (= 7 popped bubbles), and every 90th second too
	var difficulty = Math.floor((score+sField.solidScore)/7+time*2/pushRate);
	// 10 seconds is the fastest interval for adding contents (as if it wouldn\'t be hard enough with one minute)
	var pushDiff = (difficulty>pushRate-10)? pushRate-10 : difficulty;
	// Check if there are content left on the third row; if not, more rows should be added
	var tooFew = true;
	var finished = false;
	for(var j=0;j<bubblesPerRow &&!finished;j++) {
		if(grid.cells[2][j].v!=null) {
			tooFew = false;
			finsished = true;
		}
	}
	// Used for finding the lowest content and the amount of rows that should be added
	var rows = 2;
	var lowest=0;
	finished = false;
	if((pushCount>(pushRate-pushDiff)||tooFew) && !shooting) {
		// Find the lowest content
		for(var i=amountRows-1;i>=0&&!finished;i--) {
			for(var j=0;j<bubblesPerRow&&!finished;j++) {
				if(grid.cells[i][j].v!=null) {
```

```
                              lowest = i;
                              finished = true;
                        }
                  }
            }
            // Add bubbles on more rows if there are less bubbles
left. This enables the generation of new elements, so it is
rewarding
            if(lowest<Math.floor(amountRows/4))
                  rows = 4;
            generateContentOnRows(rows);
            pushCount=0;
      }
      // Keyboard control on Reader
      if(keyDown) {
            var a = reader.a;
            if(key=="left") {
                  a-=Math.PI*2/180;
            }
            else if(key=="right") {
                  a+=Math.PI*2/180;
            }
            if(a<minAngle) a = minAngle;
            else if(a>maxAngle) a = maxAngle;
            reader.a = a;
      }
      renderGame();
      if(gameOn)
            setTimeout(gameLoop,1000/fps);
}
```

Bilaga 10

```
var prevI;
var prevJ;
function bubbleHitTest(b) {
	// Step 1: Check if bubble hits, place it and check if bubbles should be popped
	var hitting = false;
	// If the bubble goes in opposite direction or over the limits, it is probably due to hacking,
	// but in case of a mysterious bug, we simply remove the bubble silently
	if(b.y > reader.by-br) {
		b.Remove();
		return true;
	}
	// Check if the bubble hits the "roof"
	if(b.y < margin) {
		hitting=true;
	}
	var i = grid.RowIFromY(b.y);
	var j = grid.ColumnJFromXI(b.x+br,i);
	if(grid.cells[i][j].v != null) {
		hitting = true;
	}
	if(hitting) {
		// Check game over (true, if the bubble is too close to the border or the reader)
		if(i==amountRows-1 || grid.cells[prevI][prevJ].v!=null) {
			gameOver();
			return true;
		}
		// Place the bubble at the previous grid position
		b.x = grid.cells[prevI][prevJ].x;
		b.y = grid.cells[prevI][prevJ].y;
		grid.cells[prevI][prevJ].v = b;
		// Check if popable and pop
		var combs = getCombinations(prevI,prevJ);
		if(combs.length>2) {
			for(var ix=0;ix<combs.length;ix++) {
```

```
      grid.cells[combs[ix].i][combs[ix].j].v.Pop();
                  grid.cells[combs[ix].i][combs[ix].j].v=null;
            }
            // Set a bonus for every 6th pop in a cluster
            score+=Math.floor(combs.length/6);
            // Any bonus drawer object should receive a value
here.

            // Step 2: Find unlinked bubbles, repeat from step
4 until there sertainly is no more unlinked
            var allfinished = false;
            var iterations = 0;
            while(!allfinished) {
                  // Check if there are bubbles unlinked
                  // The algorithm starts by finding all
bubbles linked to the "roof".
                  // Then it is easy to find the released
bubbles.
                  var links = getNonReleasedContent();
                  var linked;
                  for(var ix=1;ix<amountRows;ix++) {
                        for(var jx=0;jx<bubblesPerRow;jx++) {
                              if(grid.cells[ix][jx].v!=null) {

      if(grid.cells[ix][jx].v.type=="bubble") {
                                          linked = false;
                                          for(var
kx=0;kx<links.length && !linked;kx++) {

      if(links[kx].i==ix && links[kx].j==jx)
                                                linked =
true;
                                          }
                                          if(!linked) {
                                                // Pop the bubble
if it is not linked to the "roof"

      grid.cells[ix][jx].v.Pop();

      grid.cells[ix][jx].v=null;
                                          }
                                    }
```

```
                              }
                        }
                  }
                  // Check if there are elements being
released. In case there are elements on top of each other, the
procedure is looped
                  var releasedElements = new Array();
                  var gi, gj, released;
                  var prevRel = 0;
                  var finished = false;
                  while(!finished) {
                        for(var ix=0;ix<elements.length;ix++) {
                              if(elements[ix].released)
continue;
                              gi = elements[ix].gi;
                              gj = elements[ix].gj;
                              // It is impossible for an
element to reach the bottom of the grid
                              if(gi<amountRows-2) {
                                    // Go through the contents
below the element and check if it has been released
                                    released = true;
                                    for(var jx=-1;jx<3 &&
released;jx++) {

      if(grid.cells[gi+2][gj+jx].v!=null)
                                                released = false;
                                    }

                                    if(released) {

      releasedElements.push(grid.cells[gi][gj].v);
                                          // Mark the element as
released, to prevent double counting

      grid.cells[gi][gj].v.released = true;
                                          // Empty the whole
element space
                                          for(var
kx=0;kx<3;kx++) {

      grid.cells[gi][gj+kx].v = null;

      grid.cells[gi+1][gj+kx-1+gi%2].v = null;
```

```
                                        }
                                    }
                                }
                            }
                            // Check if new elements have been
released
                            if(releasedElements.length>prevRel) {
                                prevRel =
releasedElements.length;
                                iterations++;
                            }
                            else {
                                finished = true;
                                // Put the elements into another
function that animates the releases correctly
                                if(releasedElements.length>0)

      releaseElements(releasedElements);
                            }
                        }
                        // If no new elements have been released,
there is no need for more checks.
                        if(iterations==0)
                            allfinished = true;
                        else iterations = 0;
                    }
                }
      }
      prevI = i;
      prevJ = j;
      return hitting;
}
```

Bilaga 11

```
// Not a true event handler but acts as one
function gameOver() {
	// End game
	gameOn=false;
	// Dispatch events
	container.removeEventListener("mousemove",mouseMove,false);
	container.removeEventListener("click",onClick,false);
	document.removeEventListener("keydown",onKeyDown,false);
	document.removeEventListener("keyup",onKeyUp,false);
	// Play game over sound
	sfx["gameover"].play();
	// Show game over text
	gameOver_div.style.visibility="visible";
	// Check high score
	var s_check_xhr = new XMLHttpRequest();
	s_check_xhr.onreadystatechange = function() {
		if (s_check_xhr.readyState == 4 && s_check_xhr.status
== 200){
			var resp = s_check_xhr.responseText;
			//alert(resp);
			resp = resp.substring(10,resp.length-11);
			resp = resp.split(",");
			if(resp[0]!=0) {
				if(resp[1]!=0) {
					var emailHtml = '';
					//Draw Submit
					if(resp[1]==1) {
						// Draw email-field
						var emailHtml = 'Email address
(if you wish to win a price if you stay as the <i>Player of the
month)</i>:<br/>'
						+'<input type="text" id="pemail"
value="'+pemail+'"/><br/>';
					}
					highscore.innerHTML='<div
style="background-color:#FFFFFF;">'
					+'<h2>Congratulations, you have made it
to the High Score Table!</h2>'
					+'Score:
'+(sField.solidScore+score)+'00   Position:
'+resp[1]+'<br/><br/>'
```

```
                              +'Name:<br/><input type="text"
id="pname" value="'+pname+'"/><br/>'+emailHtml
                              +'<button
onclick="showHighScore(true);">Submit Score</button>'
                              +'<button
onclick="showHighScore(false);">Cancel</button></div>';
                              highscore.style.visibility="visible";
                        }
                        else {
                              // Draw HighScore table
                              showHighScore(false);
                        }
                  }
                  else resetGame("No Cheating!");
            }
      }
      setTimeout(function() {
            gameOver_div.style.visibility="hidden";

      s_check_xhr.open("POST","rfid_game_plugin/session_controller.
php",true);
            s_check_xhr.setRequestHeader("Content-
type","application/x-www-form-urlencoded");
            var
code=""+v+startGame.toString()+updateScore.toString()+gameLoop.toS
tring()+resetGame.toString()+playAgain.toString()+
            gameOver.toString()+bubbleHitTest.toString();
            var args =
"status=2&score="+encodeURIComponent(score)+"&pops="+encodeURIComp
onent(bPops)

      +"&rels="+encodeURIComponent(eRels)+"&energy="+encodeURICompo
nent(energy)+"&code="+encodeURIComponent(code);
            //var args =
{status:3,score:score,pops:bPops,rels:eRels,code:code};

      //alert('status=0&score='+score+'&pops='+bPops+'&rels='+eRels
);
            //alert(code);
            s_check_xhr.send(args);
      },3000);
      // Show eventual form. It just sends the name. The score will
already be on the server.
```

```
      //alert(document.body.outerHTML);
      /*
      xmlhttp.open("POST","ajax_test.php",true);
      xmlhttp.setRequestHeader("Content-type","application/x-www-
form-urlencoded");
      xmlhttp.send("0,"+id+","+score+"code");
      */

}
```

Bilaga 12

```
// r is Boolean for result, h is numeric Boolean for high score to
show, that is monthly or all times best.
function showHighScore(r,h) {
      ctx.clearRect(0,0,osize+margin*2.1,osize+margin*2.1);
      if(h==null) h=0;
      var s_check_xhr = new XMLHttpRequest();
      s_check_xhr.onreadystatechange = function() {
            if (s_check_xhr.readyState == 4 && s_check_xhr.status
== 200){
                  var resp = s_check_xhr.responseText;
                  resp = resp.substring(10,resp.length-11);
                  var menuHandler = "backToMenu();";
                  var playAgain = '<img src="'+ebgr+'all-times.png"
onclick="showHighScore(null,1);"'
                  +'onmouseover="this.src=\''+ebgr+'all-times-hover-
on.png\';"'
                  +'onmouseout="this.src=\''+ebgr+'all-
times.png\';"'
                  +'style="width:'+s*0.16+'px;
position:absolute;top:2%;left:1%;cursor:pointer;"/>';
                  if(h==1) playAgain = '<img src="'+ebgr+'this-
month.png" onclick="showHighScore(null,0);"'
                  +'onmouseover="this.src=\''+ebgr+'this-month-
hover-on.png\';"'
                  +'onmouseout="this.src=\''+ebgr+'this-
month.png\';"'
                  +'style="width:'+s*0.16+'px;
position:absolute;top:2%;left:1%;cursor:pointer;"/>';
                  var menuImgSrc = ebgr+'back-text.png';
                  if(r!=null) { menuHandler = "playAgain(false);";
                        playAgain = '<img width="'+s*0.18+'"
src="'+ebgr+'play-again-button.png" onclick="playAgain(true);"'

      +'style="position:absolute;top:12%;left:1%;cursor:pointer;"/>
';
                        menuImgSrc = ebgr+'menu-text.png';
                  }
                  highscore.innerHTML='<div style="background-
color:#EFEBEA;height:'+s*0.1+'px;
width:100%;position:absolute;top:0px;right:0px;">'
                  +'<div style="text-align:center;font-
size:24px;position:absolute;top:25%;">HIGHEST
SCORE</div>'+playAgain
```

```
                +'<img width="'+s*0.16+'" src="'+menuImgSrc+'"
onclick="'+menuHandler+'"'

     +'style="position:absolute;top:23%;right:1%;cursor:pointer;"/
></div>'


                +'<div class="scrollable"
style="overflow:auto;position:absolute;top:'+s*0.12+'px;left:'+s*0
.01+'px;width:'+s*0.75
                +'px;height:85%;font-size:'+12/380*s+'px;">'
                +'<table style="color:#000000;width:100%;">'
                +'<tr><td style="text-
align:center;">PLACE</td><td>NAME</td><td style="text-
align:center;">SCORE</td>'
                +'<td style="text-
align:center;">DATE(D.M.Y)</td></tr>'+resp+'</table></div>';
                highscore.style.visibility="visible";
           }
     }
     s_check_xhr.open("POST","rfid_game_plugin/session_controller.
php",true);
     s_check_xhr.setRequestHeader("Content-type","application/x-
www-form-urlencoded");
     var
code=""+v+startGame.toString()+updateScore.toString()+gameLoop.toS
tring()+resetGame.toString()+playAgain.toString()+
     gameOver.toString()+bubbleHitTest.toString();
     var args =
"status=4&score="+h+"&pops=0&rels=0&energy=0&code="+encodeURICompo
nent(code);
     if(r!=null) if(r) {
           pname = document.getElementById("pname").value;
           var email_dom = document.getElementById("pemail");
           // Get the email value if it exists, else keep any
previous value stored and send an empty string to the server
           var tempEmail = "e";
           if(email_dom!=null) {
                tempEmail = pemail = email_dom.value;
           }

           args =
"status=3&score="+h+"&pops=0&rels=0&energy=0&name="+pname+"&email=
"+tempEmail+"&code="+encodeURIComponent(code);
     }
     s_check_xhr.send(args);
```

```
        menuDisabler.style.visibility="visible";
        menuDisabler.style.opacity=0.4;
        mainMenu.style.zIndex=-1;
    }
```

Bilaga 13

```
// Custom Drop Down Menu
/*
Author: Markus Silvennoinen
Started:         2014.03.02
Last edit:       2014.03.17
*/
// Custom Drop Down Menu
// Make sure the name-attribute matches the variable name
// The container should work if it is a <div>
// values and onwards are optional attributes

// Array that stores references to the 'drop downs'. For internal use.
var ddms = new Array();
// DOM-Element container, String name, IMG-URL-String[] keys, [IMG-URL-String[] onhover, Object[] values, IMG-URL-String buttonimg]
function DropDownMenu(container, name, keys, onhover, values, buttonimg) {
	if(values==null) {
		var vals = new Array(keys.length);
		for(var i=0;i<keys.length;i++)
			vals[i] = i;
		values = vals;
	}
	if(buttonimg==null)
		buttonimg = ddmr+"arrow.png";
	this.container = container;
	this.name = name;
	this.keys = keys;
	this.onhover = onhover;
	if(onhover==null)
		this.onhover = keys;
	this.values = values;
	this.vi = 0;
	this.value = values[this.vi];
	this.foc = false;
	this.onchange = function(){}
	// Create a hidden form element for this object + draw the main appearance
```

```
	container.innerHTML = '<input type="hidden"
name="'+name+'_in" id="'+name+'_in" value="'+values[0]+'"/>'
	+'<div id="'+name+'_gr"></div>';
	this.input = document.getElementById(name+'_in');
	this.graphics = document.getElementById(name+'_gr');
	// Draw main appearance
	var html = '<div id="'+name+'_main" style="border-
style:solid;border-color:#EEE;height:10%;background-color:#FFF;'
	+'background-image:url('+keys[0]+');background-repeat:no-
repeat;background-size:100%;background-position:1% 50%;">'
	//+'<img id="'+name+'_disp" src="'+keys[0]+'" width="82%"
alt="'+keys[0]+'" style="border:solid;"/>'
	+'<img src="'+buttonimg+'" width="15%" style="margin-
left:84%;"/></div>'
	+'<div id="'+name+'_menu" style="background-
color:#FFF;border:solid;visibility:hidden;" width="100%">';
	// Draw menu appearance
	for(var i=0;i<keys.length;i++)
		html += '<img src="'+keys[i]+'" width="100%"
alt="'+keys[i]+'" onclick="'+name+'.change('+i+');"'

	+'onMouseOver="this.src=\''+this.onhover[i]+'\';"onMouseOut="
this.src=\''+this.keys[i]+'\';"/><br/>';
	html += '</div>';
	this.graphics.innerHTML = html;
	//this.disp = document.getElementById(name+'_disp');
	this.main = document.getElementById(name+'_main');
	ddms.push(this);
}

DropDownMenu.prototype.setFocus = function(foc) {
	if(foc!=this.foc) {
		this.foc = foc;
		if(foc) {
			// Set focus

	document.getElementById(this.name+'_menu').style.visibility='
visible';
		}
		else {
			// Blur

	document.getElementById(this.name+'_menu').style.visibility='
hidden';
```

```
            }
      }
}
DropDownMenu.prototype.change = function(index) {
      this.vi = index;
      this.input.value = this.values[index];
      this.main.style.backgroundImage='url('+this.keys[index]+')';
      this.value = this.values[index];
      this.onchange();
}


function whichDiv(e){
      var target=e?e.target:event.srcElement;
      for(var i=0;i<ddms.length;i++) {
            //alert(target.parentNode);
            // The current focus system does only support one level
of child elements. Parent node's parents etc. can be included if
necessary.
            if(target.parentNode==ddms[i].main ||
target==ddms[i].main)
                  ddms[i].setFocus(true);
            else
                  ddms[i].setFocus(false);
      }
}
document.onclick=whichDiv;
```

Bilaga 14

```
// A graphics scaler engine based on open-source graphics editing
software
// Source: http://stackoverflow.com/questions/2303690/resizing-an-
image-in-an-html5-canvas
// Author: "syockit"
// Customization: Markus Silvennoinen


function lanczosCreate(lobes){
  return function(x){
    if (x > lobes)
      return 0;
    x *= Math.PI;
    if (Math.abs(x) < 1e-16)
      return 1
    var xx = x / lobes;
    return Math.sin(x) * Math.sin(xx) / x / xx;
  }
}


//elem: canvas element, img: image element, sx: scaled width,
lobes: kernel radius
function thumbnailer(elem, img, sx, lobes){
    this.canvas = elem;
    elem.width = img.width;
    elem.height = img.height;
    elem.style.display = "none";
    this.ctx = elem.getContext("2d");
    this.ctx.drawImage(img, 0, 0);
    this.img = img;
    this.src = this.ctx.getImageData(0, 0, img.width, img.height);
    this.dest = {
        width: sx,
        height: Math.round(img.height * sx / img.width),
    };
    this.dest.data = new Array(this.dest.width * this.dest.height
* 3);
    this.lanczos = lanczosCreate(lobes);
    this.ratio = img.width / sx;
    this.rcp_ratio = 2 / this.ratio;
    this.range2 = Math.ceil(this.ratio * lobes / 2);
```

```
    this.cacheLanc = {};
    this.center = {};
    this.icenter = {};
    process1(this, 0);
}

function process1(self, u){
    self.center.x = (u + 0.5) * self.ratio;
    self.icenter.x = Math.floor(self.center.x);
    for (var v = 0; v < self.dest.height; v++) {
        self.center.y = (v + 0.5) * self.ratio;
        self.icenter.y = Math.floor(self.center.y);
        var a, r, g, b;
        a = r = g = b = 0;
        for (var i = self.icenter.x - self.range2; i <=
self.icenter.x + self.range2; i++) {
            if (i < 0 || i >= self.src.width)
                continue;
            var f_x = Math.floor(1000 * Math.abs(i -
self.center.x));
            if (!self.cacheLanc[f_x])
                self.cacheLanc[f_x] = {};
            for (var j = self.icenter.y - self.range2; j <=
self.icenter.y + self.range2; j++) {
                if (j < 0 || j >= self.src.height)
                    continue;
                var f_y = Math.floor(1000 * Math.abs(j -
self.center.y));
                if (self.cacheLanc[f_x][f_y] == undefined)
                    self.cacheLanc[f_x][f_y] =
self.lanczos(Math.sqrt(Math.pow(f_x * self.rcp_ratio, 2) +
Math.pow(f_y * self.rcp_ratio, 2)) / 1000);
                weight = self.cacheLanc[f_x][f_y];
                if (weight > 0) {
                    var idx = (j * self.src.width + i) * 4;
                    a += weight;
                    r += weight * self.src.data[idx];
                    g += weight * self.src.data[idx + 1];
                    b += weight * self.src.data[idx + 2];
                }
            }
```

```
            }
            var idx = (v * self.dest.width + u) * 3;
            self.dest.data[idx] = r / a;
            self.dest.data[idx + 1] = g / a;
            self.dest.data[idx + 2] = b / a;
        }

        if (++u < self.dest.width)
           process1(self, u);
        else
           process2(self);
    };
    function process2(self){
        self.canvas.width = self.dest.width;
        self.canvas.height = self.dest.height;
        self.ctx.drawImage(self.img, 0, 0);
        self.src = self.ctx.getImageData(0, 0, self.dest.width,
    self.dest.height);
        var idx, idx2;
        for (var i = 0; i < self.dest.width; i++) {
            for (var j = 0; j < self.dest.height; j++) {
                idx = (j * self.dest.width + i) * 3;
                idx2 = (j * self.dest.width + i) * 4;
                self.src.data[idx2] = self.dest.data[idx];
                self.src.data[idx2 + 1] = self.dest.data[idx + 1];
                self.src.data[idx2 + 2] = self.dest.data[idx + 2];
            }
        }
        self.ctx.putImageData(self.src, 0, 0);
        self.canvas.style.display = "none";
    }
```